PDS 3 Information Model Specification

Compiled Responses from

Working Group and Discipline Node Review

April through August 2008

August 28, 2008

Table of Contents

20. PDS Specification - E. Guinness Review 1 - Response 2 - Final

1. PDS3 Specification - D. Simpson Review 1 - Response 3

Hi all,

This resend includes the latest comments regarding Dick's first review of the PDS specification document.

080417 - Review of responses from #12 to end - WG
080410 - Review of responses from #1 to #11 - WG
General principles developed
080407 - Response to review comments - Steve
080406 - Review comments - Dick

thanks,
steve

General Principles
---
1) When a difference exists between the Standards Reference and the Planetary Science Data Dictionary, the Data Dictionary will have precedence. For ease of use, the team will be using the Tool Data Dictionary (PDSDD.FULL) as a reference text. This file is extracted periodically from the Data Dictionary tables.

2) Anomalies that are identified and that are associated with the model will be noted and included in the PDS Information Model Specification document. Inconsistencies that exist between the Standards Reference and the Data Dictionary will be noted in a Standards Document Inconsistencies document.

3) Where a modeling construct exists in the implemented database but is not currently documented in the Stds Ref or PSDD, it will be flagged in some way.

4) Suggestions for reorganizing the spec document are being compiled but tabled until the general review is complete. New sections might be needed to illustrate specific viewpoints (e.g. structural hierarchies)

Review of responses - 080410
---

1) During the review we started comparing the model in the specification to PDSDD.FULL (1R69) and found several inconsistencies. These were fixed.

2) Question - Do we or do we not include PSDD as an optional element? PSDD is now included - 080417

3) Referenced_From in Class tables - After further discussion we agreed to the following.

a) Limit Reference_From to association references.

b) All associations, inherited or "native", should be included. (TBD)

4) After considerable discussion regarding "uses_pointer", we considered classifying the data object description classes into two categories under two abstract classes, those that use and those that don't use pointers. The next version will demo this. Actually some classes remaining in the "uses" class do not have pointers (e.g. Text) but logically would seem to. This last point needs further discussion.

5) Object vs Subobject - Here the discussion evolved to whether in section 8 we are modeling structures or descriptions of structures. There is general agreement that we are modeling descriptions of structures. The point then raised by Anne, is how to we model the ODL concept of subobject, as in a Column object is a subobject (or nested) within a Table Object. This ODL construct implies that a Column Object is a structural component of a Table. The current suggestion is to name the Table/Column association so that this structural relationship is more explicit. E.g. Instead of "has_Column" the association could be named something like "Structural_Component_Of".

6) Removed Tagged_Container since Container is a component.

7) Move DATA_OBJECT_DESCRIPTION out of section 8? We did not finished the discussion, however there is some consensus to leave this class in section 8. The addition of a new "no pointer" parent class lends support to this approach.

8) There was a consensus to model Gazetteer Columns as a single class with multiple values for Name, as exists in version L. Same for Index Table. Will wait for Dick's response to finalize.

9) Will wait for PDS4 to model dependencies like Bit_Column needing start_byte from Column and the pointer from Table.

-- 080417 --

10) History - Will leave History_Group in model. Will change pointer card to 1 for History since History is used to point to the "history" data section in the data file. Will add PSDD to History. Will add specific keywords to History Group.

11) In the specification, a cardinality of none will be replaced by 0.

12) Deleted Time_Series since it is not defined as an explicit object. Add something to narrative to explain this.

13) Discussed the difference between class hierarchy (e.g. Table and Spectrum) and a hierarchical structure (e.g. Table and Column). Class hierarchy is specifically modeled and shown in spec document at the beginning of each section. Structure hierarchy is an association. The association name, description, and type can indicate a structural hierarchy. Consider a new section or subsection in the document specifically for structure hierarchies.

14) Document as an anomaly, the inheritance of table_storage_type by spectrum and setting of cardinality to 0.

15) Clarify in document that if Cardinality = 0 then the semantics are that there is no keyword as well as no value.

16) Qube tabled until formal review.

At 02:24 PM 4/7/2008, J. Steven Hughes wrote:

Hi Dick,

Resend with ccs as suggested by Dick.

Below are my responses.

At 09:52 AM 4/6/2008, Dick Simpson 650-723-3525 wrote:

Steve, Anne, Mitch:

Following are comments on PDS3 Specification, version 0.070916k, mostly on Section 8. These are comprehensive but not exhaustive. I left a couple blanks (FILE and SPECTRAL_QUBE), planning to return after seeing how the simpler issues flew. There have probably been discussions and decisions on some of these questions before; I continue to run behind.

1. Referenced From. Errors in the Referenced From sections of the class definition tables are:

ALIAS: Add COLUMN and FIELD; maybe more?
BIT_COLUMN: Remove COLLECTION; add COLUMN, ARRAY
BIT_ELEMENT: Add COLLECTION
COLLECTION: Add ARRAY
COLUMN: GAZETTEER_TABLE, INDEX_TABLE, PALETTE, SERIES, and SPECTRUM
have columns; but some are restricted, so fixing this may depend on how some other problems are handled (such as the ones

identified for GAZETTEER_TABLE and INDEX_TABLE, below).
CONTAINER: Add CONTAINER, SERIES, SPECTRUM, and TABLE
FILE: Add VOLUME. Is TAGGED_DATA_OBJECT a useful answer here?

Mostly good calls and the problems have been fixed.

Issue: Columns are not Referenced from Series and Spectrum since Series and Spectrum are subclasses of Table. The "Reference from" field is strictly an index of where the term was referenced and more specifically the usage defined. One could argue that an inherited term is a referenced. It could also be argued that it is not. I chose the later but am open to the other option.

Issue: Volume is not specified in Appendix A as an optional object of File.

2. Association: Errors in the Association sections of the class definition tables are:

ARRAY: Add has_array, has_bit_element, and has_collection
CONTAINER: Add has_container
SERIES: Add has_container
SPECTRUM: Add has_container
TABLE: Add has_container

Fixed implicitly as fixes for Comment 1 (above) were applied.

*** Modeling Issue - Currently "similar" associations have been modeled in two different ways in the specification and not always consistently. For example, "has_Column" in a generic association that could apply to Table, Series, Gazetteer_Table, etc. However, as the association become more specific (i.e. the description or cardinality changes) then handling the additional specificity forces modeling decisions. The class hierarchy situation for example, (e.g. Table -> Series and both having has_Column) is pretty straight forward and one can generally make the association more specific (restrict) as needed, if needed.

However has_Element, an association for both ARRAY and COLLECTION should probably be modeled as two separate associations. The association between GAZATTEER and COLUMN (has_Gazetteer_Column) is a good example. I intend to work toward making these associations consistent but wanted to warn everyone.

3. Inherited Association: In a few cases, Inherited Association "uses_pointer" is insufficient to locate you in the object. For example, if you're in a BIT_COLUMN you need to inherit both the pointer to the TABLE and the START_BYTE of the COLUMN; only the pointer is inherited and therefore available.

Bit_Column is not a subclass of Table or Column. So no attributes can be inherited from either. Bit_Column inherits Pointer from Data_Object_Description, the parent Class of all three.

This is more of a "dependency" type of relationship. I suggest that this is a "late" PDS4 data modeling issue to be addressed once we complete the core model.

4. Is class vs sub-class given by "hierarchy"

Yes.

whereas object vs sub-object is given by Referenced_From and Association?

No. "Referenced from" was something I added as a simple index to help determine where a term has been used. There were no modeling "semantics" intended however the references are currently only those involving an "association". If useful, it could be renamed to something like "associated with" and restricted to references of the class in an "association".

The characteristics of Object vs Sub_Object are strictly determined by the class and subclass definitions and nothing more or less.

If so, I think I understand; but this is a bit murky and it would be helpful somewhere in the document to have this difference explained. Paragraph 2 at the beginning of Section 8 only covers half of the story.

We should explain or change the meaning of "Referenced from".

5. ALIAS. ALIAS can supposedly be used to give alternate names to approved elements and objects. I have no problem seeing how this would be done for objects; how would it work with an element?

Good point. This is addressed in anomaly "010_080229_003_MissingIDs_LinkClasses" which once parsed, says that we need to allow a set of standard identification elements for all classes. In particular the text suggests using the W3C Dublin Core element "alternate". This element would allow all instantiated objects to have alternate names (i.e. identifiers, acronymns, name, etc.)

As for alternate names for elements, "alternate" added to the data dictionary would solve the problem. We currently have "alias".

Note that alternate names are often associated with a time frame and source.

6. COLLECTION: Why is there no tagged_Collection?

Added.

7. CONTAINER: Why is there no tagged_Container?

Added.

Removed since it is a component - 080417

8. COLUMN. Correct Association by moving uses_pointer to Inherited Association; add has_bit_column and has_alias.

Depending on how some of the more complicated questions with TABLE and its cousins are resolved (below), it may be necessary to add those classes to the Referenced From section of the definition table.

Yes, we need to address this question after other questions are settled.

9. DATA_OBJECT_DESCRIPTION. I think it makes more sense to move this to section 9. In terms of "hierarchy," all you need to say is that section 8 is simply an explosion of the single data_object_description line in section 9.

Could be done however the current scheme focuses on having a parent class and its subclasses grouped into a section. This is a logical grouping and results in the class hierarchies.

Something that we have not discussed before that might help are the concepts of abstract and concrete classes. Abstract classes can not be instantiated. DATA_OBJECT_DESCRIPTION is a good example of a class that might not make sense to instantiate. Concrete classes are things like Table, etc.

As the parent or super class, DATA_OBJECT_DESCRIPTION sets a basis on which all the subclasses are built. Of course there is no strict reason that a parent class has to be abstract, but when it is as in this case, it also helps "classify". For example, our Catalog_Description is different by one attribute and distinguishes the classes used to describe digital objects from those that describe non-digital objects.

In the class definition table, would it be better to show Referenced From "N/A"? Data_object_description is not something we will see in a label.

Will revisit this after we settle Reference_From.

10. GAZETTEER_TABLE and INDEX_TABLE. If we allowed object/keyword dependencies, these two tables could both be special cases of TABLE and the model would be simpler. Because dependencies are not allowed in PDS3, I think we're stuck with a couple more levels of branches in the model. Not only are GAZETTEER_TABLE and INDEX_TABLE required, but their required and optional columns must also be included explicitly in the model (because they have their own requirements). GAZETTEER_TABLE and INDEX_TABLE seem to have been handled differently; I don't know why. I seems logical to deal with them in the same way.

Yes, this was corrected in Version L so they are consistent. Although the overall modeling approach is different. Needs discussion.

For GAZETTEER_TABLE, we have the following required columns. Two already appear in the spec at 8.14 and 8.16; I'm not sure why the others weren't included. I suggest we skip GAZETTEER_COLUMN (8.13) as unnecessary clutter and go directly to

GAZETTEER_TABLE_TARGET_NAME_COLUMN
GAZETTEER_TABLE_SEARCH_FEATURE_NAME_COLUMN
GAZETTEER_TABLE_DIACRITIC_FEASTURE_NAME_COLUMN
GAZETTEER_TABLE_MINIMUM_LATITUDE_COLUMN
GAZETTEER_TABLE_MAXIUM_LATITUDE_COLUMN
GAZETTEER_TABLE_CENTER_LATITUDE_COLUMN
GAZETTEER_TABLE_MINIMUM_LONGITUDE_COLUMN
GAZETTEER_TABLE_MAXIMUM_LONGITUDE_COLUMN
GAZETTEER_TABLE_CENTER_LONGITUDE_COLUMN
GAZETTEER_TABLE_LABEL_POSITION_ID_COLUMN
GAZETTEER_TABLE_FEATURE_LENGTH_COLUMN
GAZETTEER_TABLE_PRIMARY_PARENTAGE_ID_COLUMN
GAZETTEER_TABLE_SECONDARY_PARENTAGE_ID_COLUMN
GAZETTEER_TABLE_MAP_SERIAL_ID_COLUMN
GAZETTEER_TABLE_FEATURE_STATUS_TYPE_COLUMN
GAZETTEER_TABLE_APPROVAL_DATE_COLUMN
GAZETTEER_TABLE_FEATURE_TYPE_COLUMN
GAZETTEER_TABLE_REFERENCE_NUMBER_COLUMN
GAZETTEER_TABLE_MAP_CHART_ID_COLUMN
GAZETTEER_TABLE_FEATURE_DESCRIPTION_COLUMN

I admit that my suggestion of abandoning GAZETTEER_COLUMN here is not entirely consistent with some of my other suggestion elsewhere; but a consistent approach would be best whatever it is.

The table in 8.15 should show attribute "columns" having value 20. The Associations section of the table must be expanded to reflect the additions above. The Referenced From section of the table must be corrected to show TAGGED_GAZETTEER_TABLE; and TAGGED_GAZETTEER_TABLE
needs to be added to section 9. For parallelism with INDEX_TABLE, we should probably run TAGGED_GAZETTEER_TABLE back through TAGGED_TABLE; I
may be missing something, however, so I'll end this thread here.

Cardinality has been changed.

I recommend the above class definitions be added to the spec as sections 8.15.1 through 8.15.20. This breaks the alphabetical ordering; but I would argue that these subclasses can appear nowhere else but under GAZETTEER_TABLE so it makes sense to group them under the parent class. We could omit them from the Table of Contents to keep the T of C simpler.

INDEX_TABLE should have the same type of substructure. There is no INDEX_COLUMN in the spec (which I would have dropped had it been there). The required and optional columns to be added are

INDEX_TABLE_NAME_COLUMN
INDEX_TABLE_PRODUCT_ID_COLUMN
INDEX_TABLE_VOLUME_ID_COLUMN
INDEX_TABLE_DATA_SET_ID_COLUMN
INDEX_TABLE_PRODUCT_CREATION_TIME_COLUMN
INDEX_TABLE_VOLUME_PATH_NAME_COLUMN
INDEX_TABLE_MISSION_NAME_COLUMN
INDEX_TABLE_INSTRUMENT_ID_COLUMN
INDEX_TABLE_INSTRUMENT_NAME_COLUMN
INDEX_TABLE_INSTRUMENT_HOST_ID_COLUMN
INDEX_TABLE_INSTRUMENT_HOST_NAME_COLUMN
INDEX_TABLE_SPACECRAFT_ID_COLUMN
INDEX_TABLE_SPACECRAFT_NAME_COLUMN
INDEX_TABLE_TARGET_NAME_COLUMN
INDEX_TABLE_PRODUCT_TYPE_COLUMN
INDEX_TABLE_MISSION_PHASE_NAME_COLUMN
INDEX_TABLE_VOLUME_SET_ID_COLUMN
INDEX_TABLE_START_TIME_COLUMN
INDEX_TABLE_STOP_TIME_COLUMN
INDEX_TABLE_SPACECRAFT_CLOCK_START_COUNT_COLUMN
INDEX_TABLE_SPACECRAFT_CLOCK_STOP_COUNT_COLUMN

These should be sections 8.24.1 through 8.24.21 in the spec; but they should not appear in the Table of Contents.

11. HISTOGRAM. Correct cardinality of items to be 1.

Good call.

12. HISTORY and HISTORY_GROUP. How can HISTORY and HISTORY_GROUP both
have Association uses_pointer none Data_Object_Pointer? How does the user know what file(s) the history is about? The "by convention" note in PDSSR is possibly helpful, but far from definitive. Why is there no tagged_History? This looks like a PDS3 anomaly/ambiguity, possibly for several reasons?

Yes, needs more discussion.

13. FILE, IMPLICIT_FILE, etc. TBD

Thanks ;-)

14. PALETTE. Correct Attribute by adding description (0..1) and name_ (0..1)

done.

15. QUBE. I would prefer Association has_Qube_Line_Suffix, has_Qube_Band_Suffix, and has_Qube_Sample_Suffix.

16. QUBE_***_SUFFIX. The attributes for QUBE_LINE_SUFFIX should be line_high_instr_sat, line_high_repr_sat, ...; "band_" should be removed. Similarly "band_band_" should be reduced to "band_" in the QUBE_BAND_SUFFIX definition and "sample_band_" should become "sample_" in the QUBE_SAMPLE_SUFFIX definition.

17. QUBE_SUFFIX. Suggest Referenced From "N/A" (see DATA_OBJECT_DESCRIPTION above).

Seems reasonable but I would like the more input before making the change.

18. SERIES. I hate to make the model more complicated, but I think SERIES should not be a sub-class of TABLE (at least in PDS3); it has additional required keywords. List this as a PDS3 anomaly, to be cleaned up in PDS4(?).

I do not see this as an anomaly. Series (and Spectrum) is one area where I think PDS3 is consistent with the object-oriented paradigm. It nicely demonstrates how PDS4 should look.

Add Attributes sampling_parameter_name, sampling_parameter_unit, and sampling_parameter_interval (1).

Was added in verion L.

Inherited Attribute name_ should have cardinality (0..1)

Fixed in version L.

What is Inherited Attribute table_storage_type?

An anomaly.

19. SPECTRAL_QUBE. TBD

Yep.

20. SPECTRUM. Like SERIES, this should be separated from TABLE.

Inherited Attribute name_ should have cardinality (0..1)

What is Inherited Attribute table_storage_type?

Actually as a subclass, fixing Series also fixed Spectrum and demos the benefits of

having a class hierarchy.

21. SPREADSHEET. Since FIELD can only be used with SPREADSHEET, we could list it under SPREADSHEET and remove it as a separate entry in the Table of Contents (see above).

Needs discussion.

Attribute name_ should have cardinality (0..1)

Fixed in Version L.

22. TABLE. Table_Binary and Table_ASCII should be removed from Subclass. Otherwise, the list is open to change based on decisions made in response to other comments.

Attribute name_ should have cardinality (0..1)

What is Attribute table_storage_type?

Done in Version L.

23. TIME_SERIES. Omit; SERIES with chronological data are adequately covered by SERIES.

Will await resolution of SERIES issues.

24. The path from class through tagged_class to something higher seems inconsistent. I have noted some specific problems above; I tried to compile a complete list, but there were so many "what ifs" that I decided to wait until clearing the decks of the current list of problems. Perhaps there will be a general resolution along the way.

Comment to be taken up again as we progress through the document.

This was very helpful. You found items I had missed and it seems that you understand the document.

Many thanks,
steve

Regards,
Dick

---

2. PDS3 Specification - D. Simpson Review 2 - Response 2b

-AUTH: Authorized

Hi all,

Resend with additional comments added. No need for additional review.

The following contains the compiled responses to Dick Simpson's cross checking review of the PDS3 Specification document.

thanks,
steve

080501 - Additional comments on NSSDC_DATA_SET_ID - R. Raugh and S. Hughes
080424 - Review of responses - WG
080423 - Response to review comments - Steve
080422 - Review comments - Dick

Review of responses - 080424 - WG - Additional comments
---
1) In general the WG agreed with the initial responses except where noted in the following.

2) Cleanup class descriptions by stripping out sentences that mention obsolete subobjects and other ODL constructs.

3) Fix Node_ID within the context of the Inventory class (data distribution nodes) to constrain the valid values to node ids that actually distribute data.

4) Add an Operations abstract class as parent for housekeeping classes and etc.

5) Agreed that that the associations to References have a cardinality of one. If no reference is available, the valid value N/A is used. Add narrative to an appropriate section of the document to explain why the use of figurative constants such as "N/A" precludes the need for cardinalities of zero.

6) Reiterated need for narrative to explain repeating groups and the merging of subobjects.

7) Add the Mission_Target relation to the PDS anomaly list for further discussion.

8) All inconsistencies between PSDD and Std Ref to be sent to Elizabeth to be considered as part of her V3.8 release.

At 10:22 PM 4/23/2008, J. Steven Hughes wrote:
Hi Dick,

Responses below.

I have denoted certain responses with a ***1 for an open issue or for further discussion, ***2 for a proposed PDS3 anomaly, and ***3 for inconsistencies in either the PSDD or the Stds Ref.

thanks,
steve

080423 - Response to review comments - Steve
080422 - Review comments - Dick

At 10:22 AM 4/22/2008, Dick Simpson 650-723-3525 wrote:
Steve:

For what it's worth, this is my review of the 16 objects I was assigned. Thanks for the help getting started yesterday; the first three are repeated but with minor mods.

Dick

- -

Files referenced:

1. PDS3 Spec 0.070916m
2. PDS SR 3.7
3. pdsdd.full 1r69
4. PDS on-line dictionary (secondary to 1r69)

General Comments:

Many object definitions in 1R69 list sub-objects in DESCRIPTION.
Not only are these usually in some very terse vocabulary but many
are also wrong. The simplest solution is to omit the DESCRIPTION
listing, since the required and optional sub-objects must be
listed under REQUIRED_OBJECT_SET and/or OPTIONAL_OBJECT_SET.

***1 I would like have a broader discussion. I agree that the class_descriptions have a lot of problems. They were either copied and pasted from the std ref or loaded directly from the PSDD. If we wish to take the time we can change them as part of our current work.

General Comments on Chapter 11:

line 2: "data is" --> "data are"

Fixed.

line -4: "Figure 5" --> "Figure 6"

Fixed.

Figure 6: where are the following?

HOUSEKEEPING
INVENTORY
INVENTORY_DATA_SET_INFO

I have not updated the figures for several weeks so they are not current. Once the "table" view of the classes start converging I will regenerate the figures.

INVENTORY:

Not clear where this fits in Figure 6.

This will be fixed when the figure is redone using the "table" view as a guide.

I believe Association has_Inventory_Data_Set_Info should have cardinality "0..*" to allow for cases where node_id has values HQ, RAD, or RS (inappropriate, obsolete, and/or obsolete nodes). If HQ, RAD, and RS do not have completed INVENTORY objects, then the values of NODE_ID should be constrained to the nodes which do have completed objects.

***1 Interesting call but this needs some discussion. Should non-Node identifiers be included as values in the Node_id field. Even if non-Node ids are included in the data dictionary the attribute in the model can specify the valid values. The real question here is, does the model have precedence over the DD or vice versa. I believe the former.

Shouldn't there be an Association with NODE?

Yes, fixed.

Why does the spec show "Referenced from none"? This must have something to do with the way the central catalog is organized and used; in normal circumstances, I would expect INVENTORY to be "referenced from" from something higher in the operational tree.

***1 Good point. Currently there is not an appropriate abstract class from which to reference Inventory. But this might be an anomaly that points to the need for an even higher level set of modeled classes for grouping these classes.

The Class Description is very cryptic. Ditto the pdsdd.full DESCRIPTION, which also says there is a required subobject INVNETTHRSHLD (which is either a very bad typo or wrong).

See comment on descriptions.

Based on my limited understanding of INVENTORY, may I suggest the following description(s)?

One INVENTORY object exists for each PDS discipline node which distributes data. INVENTORY contains a sub-object INVENTORY_DATA_SET_INFO for each data set which can be ordered.

***1 How about the following.

One INVENTORY object exists for each PDS discipline node which distributes data. It is used to associate a discipline node with the data sets which can be ordered from that node.

I prefer keeping the descriptions more abstract.

Change not made.

INVENTORY_DATA_SET_INFO:

Not clear where this fits in Figure 6.

pdsdd.full DESCRIPTION mentions sub-objects INVSIZE, INVSZTHRSHLD, INVPINSW, and INVNODEMEDIA. Only INVNODEMEDIA is listed in REQUIRED_OBJECT_SET.

The figure will be redone. The other mentioned subobjects were removed in catalog streamlining. Clearly the description was not updated then.

INVENTORY_NODE_MEDIA_INFO:

This is shown in Figure 6; but the relationships don't appear to be correct, since INVENTORY and INVENTORY_DATA_SET_INFO are missing.

The figure will be redone.

The Class Description and DESCRIPTION in pdsdd.full both mention data set cost; but there is nothing in the table to cover this type of information.

Probably a streamlining follow-through problem.

Class Description and DESCRIPTION: "set This" --> "set. This"

Fixed.

MISSION:

The following appear to be "unauthorized" attributes in the PDS3 spec (not shown in either PDS SR or 1R69):

The Mission_Host, Mission_Information, and Mission_Reference_Information subobjects were all eliminated in the model and their data elements moved to mission. This was a modeling decision. The purpose was to reverse engineer the conceptual model from the ODL implementatation.

instrument_host_id

Instrument_host_Id was moved here from the Mission_Host subobject and the Mission_Host subobject was deleted.

mission_alias_name
mission_desc

Ditto for Mission_Information.

mission_id

Deleted since this data element does not exist.

mission_objectives_summary
mission_start_date
mission_stop_date
reference_key_id

Ditto for Mission_Information and Reference_Information.

If mission_id is allowed and has cardinality "0", this cannot be a SPECIFIC object.

Good point. However mission_id is not a data element and so has been deleted.

Association: has_Mission_Host, has_Mission_I, and has_Reference are OK; but why are the values set to Instrument_Host, Data_Set, and Reference?

The values are the names of the classes with which Mission has the association.

Shouldn't they be MISSION_HOST, MISSION_INFORMATION, and MISSION_REFERENCE_INFORMATION? I think has_Mission_I is too terse.

***1 I agree. I hope that when we discuss fixing class descriptions, we also consider associations names and their descriptions.

Both PDS SR and 1r69 require MISSION_REFERENCE_INFORMATION; so cardinality cannot be "0". If there's no appropriate reference, the PDS SR section on MISSION_REFERENCE_INFORMATION says users should set REFERENCE_KEY_ID to "N/A".

Fixed.

***1 Is this consistent with all other classes with associations to Reference? Need help.

If we're adopting the convention that keyword = "N/A" is logically equivalent to cardinality 0, we should say so in Section 6.

***1 Yes, but I would not say that we have adopted the convention as stated. From the modeling perspective a value is required. The PDS convention is that the value "N/A" is to be used when no other value is appropriate. The meaning is "Not Applicable". This is probably what should be documented.

A cardinality=0 implies that the data element exists but that its value is <null>. <null> is a symbol that signifies that no value has been provided. This is different from the token "NULL" or "null" which are figurative constants like as "N/A" or "UNK". These are valid values.

If we are adopting the convention that an object definition within which all keywords are "N/A" and all subject-objects have cardinality 0 has cardinality 0 itself, then we should state that in Section 6.

See above. In the PDS, "N/A" is the name of an instrument, mission, etc so the respective cardinalities are 1.

During last week's telecon, I believe you made the argument that keyword = "NULL" was logically equivalent to cardinality 0; I don't believe that is true, since NULL implies that a value exists -- it simply hasn't been found yet. "N/A" would be a better choice.

I might have not have said it right, but <null> implies that no value has been provided. This is true in data management in general and I thought accepted in the PDS. The figurative constance "N/A", "UNK", and ("NULL" ?) would be values with special meanings.

Referenced from: It's true that CATALOG requires MISSION; but DATA_SET requires DATA_SET_MISSION, which is a different object.

This subobject has not been modeled as a class.

MISSION_HOST:

As with the DATA_SET_* objects, has this been absorbed into MISSION?

Yes.

If so, I'm not sure I understand the logic. Didn't we agree that
FIELD shouldn't be described with SPREADSHEET?

This is tricky. The difference is data modeling theory versus a particular implementation. A standard data modeling principle is that a repeating group of attributes becomes its own unique group. In the object-oriented paradigm the group is called a "class". In the relational paradigm it is a "table".

Field, Column, etc could be described within their respective classes as you suggest if they did not have attributes of their own and if they did not repeat. Since they do have attributes of their own and can repeat, they are defined as unique individual classes.

Mission_Reference_Information on the other hand has only one attribute, Reference_Key_ID and could be absorbed even though it repeats. It was not really a group but an implementation artifact to signal to the catalog loader that a many-to-many table was to be loaded.

Mission_Information has many attributes but does not repeat and so could be absorbed. It was not really a group either but also an artifact to signal to the catalog loader that the dsinfo (data set information) table was to be loaded.

Why these strange groups? The ODL templates were originally designed to load tables in the catalog. They are not good representations of the model.

Now we have made
several objects completely disappear, and the relationships among
their sub-objects have become invisible! This may be a convenient
way to save a little work; but I don't think it's healthy if we're
trying to specify PDS3 in a useful way. The MISSION* family needs to
be revisited after this is decided.

***1 Good point. I agree that the model should represent PDS3. However, a key requirement for the specification was that the model be implementation independent.

Assuming that we extract out the conceptual model and omit the implementation specific ODL artifacts, (e.g. subobject for loading catalog tables) we will want to write a

transformation routine to create ODL template files from the conceptual model. This might be the view of the model we want to include in our readers digest version. In addition since rules would have to be defined for the transformation routine, we would be creating consistent ODL, for once.

And if we decide to use XML, we write another transformation routine to create XML schemas.

Let the debates begin. 

MISSION_INFORMATION:

Same as MISSION_HOST.

Yes.

MISSION_REFERENCE_INFORMATION:

Same as MISSION_HOST.

Yes.

MISSION_TARGET:

Same as MISSION_HOST, except that this object and its relationship(s) is one that is completely invisible given the shorthand which has been adopted.

Why is MISSION_TARGET a GENERIC object when there are no optional elements or sub-objects?

Side issue: If MISSION_TARGET is to be the sub-object of something, I would choose MISSION_INFORMATION rather than MISSION_HOST. This would be the target of the mission (a limited set), whereas other flavors of target would be used when we want to associate targets with data products, data sets, and/or instruments.

***2 The Data_Set_Mission relation (subobject) essentially supplanted this relation. This relation might now be an anomaly.

NODE:

Not found in PDS SR

The node_information subobject was subsumed by the node class.

The following appear to be "unauthorized" attributes and associations in the PDS3 spec (not shown in 1R69):

discipline_desc

Deleted.

discipline_name

Moved to node class.

institution_name

Changed to node_instiution_name.

node_name

Moved to node class.

curates

This is the association to data set. It is populated by the data engineer.

da_contact_pds_users_id

Moved to node class.

distributes

This is an association to data set. It is populated by the inventory classes.

node_manager_pds_user_id
operations_contact_pds_user_id

Moved to node class.

Shouldn't there be an Association with INVENTORY (has_Inventory)?

Inventory is a housekeeping object, used to load the catalog and populate various relations that are not populated by catalog templates. e.g. Distributes.

NODE_INFORMATION:

Seems to have been subsumed into NODE; see comments for MISSION_HOST.

The subsuming seems to be imperfect. 1r69 requires keywords NODE_DESC and NODE_INSTITUTION_NAME, which have been replaced by discipline_desc and institution_name, respectively, in the spec.

Added node_desc
Deleted discipline_desc
changed institution_name to node_institution_name.

NSSDC_DATA_SET_ID:

I don't find this in the PDS3 spec, perhaps with good reason.

This appears to be a specific object definition. It is sent to NSSDC,

which responds by returning an identically named keyword. I suggest calling this one an anomaly until we figure out better what it is and does.

Good idea.

***2 NSSDC_DATA_SET_ID

>080501

A. Raugh.

No problem with the above, but there is another anomaly associated with the NSSDC_DATA_SET_ID object. It is an extremely rare one, but I first discovered it elsewhere when some tool I was using (forget what) got confused. Here's the problem:

There is an OBJECT in the pdsdd.full called "NSSDC_DATA_SET_ID".
This is also an ELEMENT in the pdsdd.full called "NSSDC_DATA_SET_ID".

If you know in advance which one you're looking for, then careful use of the index file will prevent confusion between the two. But creating an OBJECT type that ends in an ELEMENT classword is something I would have classified as a mistake. It is not, however, specifically forbidden by anything in the standards.

S. Hughes

1) The PDS/NSSDC interface is currently being redeveloped. This object had not been used for years and its replacement will be very different.
2) The modeling tool being used for the specification requires unique names across all entities, classes, attributes, and associations.

Use of DATA_SET_COLL_OR_DATA_SET_ID (in addition to DATA_SET_ID) and MEDIUMN_TYPE both appear to be unnecessary.

PALETTE:

1r69 DESCRIPTION says PALETTE is a sub-class of TABLE; but it is not shown that way in the hierarchy at the beginning of Section 8 nor in the Hierarchy at the top of its own class table.

***2 Yes, when the base table class (i.e. modeled with required keywords only) was removed, many purported subclasses of table ceased to be subclasses.

Most of the DESCRIPTION in 1r69 appears to be well-intentioned but useless. Why is ASCII vs binary dependent on whether the PALETTE is in the same file as its companion IMAGE? We have INTERCHANGE_FORMAT to handle that.

Attribute: PSDD is listed in the spec and 1r69, but not in PDS SR

***3 PALETTE includes PSDD - not consistent in 1r69 and PSDD. Question about modeling.

PARAMETERS:

1r69 shows STATUS_TYPE = PENDING? Is this correct?

***3 Std Coordinator

PERSONNEL:

Class Description: "A person which" --> "A person who"

fixed.

Most of the Attributes come from PERSONNEL_INFORMATION; I assume PERSONNEL has been merged with PERSONNEL_INFORMATION and has taken

yes

over some of the aspects of PERSONNEL_ELECTRONIC_MAIL. See comments on the MISSION* family above.

PERSONNEL_ELECTRONIC_MAIL:

PERSONNEL_ELECTRONIC_MAIL was modeled but was filtered out. It has been added back in. It is a repeating group and so can not be merged.

This object has disappeared; see PERSONNEL.

PERSONNEL_INFORMATION:

See PERSONNEL.

Yes.

1. PDS3 Specification - M. Gordon Review 1 - Response 2b

Hi all,

Resend with update. Review needed for comment 8.

The following contains the compiled responses to Mitch Gordon's cross checking review of the PDS3 Specification document.

thanks,
steve

080501 - A. Raugh and S. Hughes - Additional comments.
080424 - Review of responses - WG
080415 - Response to review comments - Steve
080415 - Review comments - Mitch

Review of responses - 080424 - WG - Additional comments
---
1) In general the WG agreed with the initial responses except where noted in the following.

2) Add some narrative to the catalog description section that describes the reason why many subobjects were merged into the parent class. This should include a brief explanation of the concept of "repeating groups". This narrative might also need to appear in other sections.

3) Add the PDS3 SPICE model to the anomaly list to be fixed in PDS4.

4) Tabled question as to whether we need to model the standard logical Volume organizations and physical layouts.

5) Expand the pointer class. The discussion of the FILE object raised the question of FILE pointers for both standard and combined-detached labels. It was recognized that the Explicit FILE object uses the FILE_NAME keyword to point to the file and not the standard data object pointer. Both will be modeled so that the distinction is clear. Structure pointers are not considered part of the model since they are directives for processing the label file.

6) Tabled the FILE subclass question. The discussion of the FILE object raise the question of the different file types and whether they should be explicitly modeled as subclasses. Mitch has identified six types of files. The WG will review Mitch's work and then decide whether FILE should have subclasses in the model. Mitch's model is included below.

7) 080501 - History_Group removed and History attributes set to PSDD as specified. A. Raugh - HISTORY object describes a section of the data file (pointed to by the ^HISTORY pointer) that LOOKS like ODL, ***but is not actually part of the PDS label***. Inside the bytes of the HISTORY object itself, PDS rules do not apply. And note that it is another anomaly that the HISTORY object (and thus the History class) has neither required nor optional elements or sub-objects. It is, actually, just an elaborate pointer to a text file segment.

8) 080501 - Discussion needed - The cardinality of has_Index_Columns must have a minimum of 1, because the only column that is absolute, positively required in an INDEX_TABLE is at least one that provides the file location.

9) 080501 - Add anomaly on Index_Table and the dependencies between suggested columns.

At 11:32 AM 4/23/2008, J. Steven Hughes wrote:

Hi Mitch,

Responses below.

I have denoted the responses with a ***1 for an open issue or for further discussion and ***2 for a proposed PDS3 anomaly. I have added ***3 for inconsistencies in either the PSDD or the Stds Ref that should be captured in another document. Mitch has listed quite a few.

thanks,
steve

At 06:25 AM 4/15/2008, Mitch Gordon wrote:

Hi Steve,

I've completed my run through with occasional excursions beyond my brief. My comments and questions are below. I rearranged my assigned objects to match the order in which they appear in the specification. Sorry if that makes it more complicated to compile the results.

Cheers,

Mitch

==========

Files Referenced:

1. PDS Information Model Specification (HTML format) Version 0.070916m
2. Standards Reference 3.7, Mar 20, 2006
3. PSDD: pdsdd.full file 1r69

==========

I was not assigned Catalog, but went back to it because I have a couple of catalog sub-classes which do not inherit anything from the parent. Now I'm confused about the Catalog description - guess I shouldn't have wandered outside my assigned objects.

> Catalog has the Association "has_Catalog_Data_Set" with
> cardinality of 1..*, but has the attribute "data_set_id"
> with cardinality of 0..1. Shouldn't these two items have
> the same cardinality?

***1 Yes, they should have the same cardinality and I will go with your suggestion. I am assuming that we are using the current volume model that allows more than one data set on a volume. Also I am assuming that the nodes have enforced this viewpoint and allowed multiple data set ids. PPI will probably be a good reviewer.

Five of my assigned objects do not appear in the specification:

> EVENT_SPATIAL - presumably needs to be added to section 11?

Anne also had some event objects. Here is my response to her.

They were designed for V1.0 and seldom if ever used. The catalog streamlining effort seemed to provide the coup de grace. If the team thinks we need to model them I will, otherwise we can just talk to them.

INSTRUMENT_HOST_INFORMATION and
INSTRUMENT_HOST_REFERENCE_INFO
- these have been subsumed into Instrument_Host. All of their keywords were retained in the process.

INSTRUMENT_INFORMATION and INSTRUMENT_REFERENCE_INFO
- these have been subsumed into Instrument. All of their keywords were retained in the process.

===== INSTRUMENT & INSTRUMENT_HOST:

As I recall, during the last telecon we said we'd show all involved subclasses in "Referenced From" (e.g., for Column, we'd list Series and Spectrum, not just Table from which they inherit Column). If that is correct, than for both INSTRUMENT & INSTRUMENT_HOST, we need to add to "Referenced From": Identification_Data_Elements--Ancillary, Identification_Data_Elements--Earthbase, Identification_Data_Elements--Spacecraft

This is the inherited association issue that has a TBD fix.

===== INSTRUMENT

I'm not sure about cardinality = 1 for "has_Instrument_I".

From Section14:
"This one (I suggest we change 'one' to 'single') many-to-many relationship is used for both Instrument_Host and Instrument relationships with Data_Set."

The analogous entry for INSTRUMENT_HOST, "has_HOST_I", has a cardinality of 1..* which seems more appropriate. If the cardinality is correct for Instrument, it may imply that

it is incorrect for Instrument_Host. It seems likely that
one of them is wrong. Your call.

Since an instrument can have many data sets, it should be 1..*. We are then assuming that at least one data set exists for each instrument.

I also fixed the cardinality on instrument_host_id. Should have been 1..*.

===== INSTRUMENT_HOST

INSTRUMENT_HOST_TYPE shows three possible values: EARTH
BASED,ROVER,SPACECRAFT. The PDSDD also allows DATA BASE.

Fixed.

"Referenced from" includes "SPICE_Package". I can't find
"SPICE_Package" in the specification, nor can I find a
SPICE entry that has an Association with INSTRUMENT_HOST.

I have removed this association. It was part of the attempt to model what Chuck wanted and should have been associated with a SPICE_Data_Set. Since we do not have data set subclasses in PDS3, this can not be done now.

Something is broken, but I'm not sure what. There are
SPICE kernels that are INSTRUMENT_HOST specific and some
that are INSTRUMENT specific. SPICE is not addressed in
the INSTRUMENT specification (no Referenced From), and the
Referenced From/Association connection identified in
INSTRUMENT_HOST seems incorrect or incomplete.

===== FIELD

The specification attributes match the PSDD list, but
there are errors in both source documents.

***3 When we resolve the following differences I will update the model.

Why is the status of Field "PENDING" in the PDSDD?

Disconnects between the StdRef & PSDD. In this case, it appears that neither is actually correct. Optional keywords in PSDD, but not in StdRef

FIELD_DELIMITER
MISSING_CONSTANT - this is mentioned in the StdRef narrative.

Optional keywords in StdRef but not in PSDD (use of these is encouraged in the StdRef narrative)

VALID_MAXIMUM
VALID_MINIMUM

The StdRef has a typo (refers to the element ITEM rather than ITEMS)in subparagraph 3 of the narrative.

===== FILE

"Referenced from" includes "Directory_Physical". I can't find "Directory_Physical" elsewhere in the specification.

I have removed the association.

Why is the cardinality of "uses_pointer" set to " None"? I would have expected 0..* I have the same question for IMPLICIT_FILE and IMPLICIT_FILE_ATTACHED.

***1 My interpretation was that the attribute FILE_NAME is used to identify the file. IMPLICIT_FILE* does not even provide a file name.

The specification attributes match the PSDD list, but there are inconsistencies between the source documents. Required and Optional are a mess. The specification has problems because of the inconsistencies between the sources and because the determination of whether a specific attribute is required or optional depends not only on the file object, but on the value of the RECORD_TYPE keyword and the type of label used.

Here's a summary of the disconnects:

According to the StdRef, only RECORD_TYPE is required for every FILE object regardless of label type. FILE_RECORDS, LABEL_RECORDS, RECORD_BYTES may be required depending on label type - see table in Appendix A15. FILE_NAME is not included in the table although it appears to share some restrictions with FILE_RECORDS. FILE_NAME is only listed as optional with a footnote that it is "...required in minimal detached labels..." (the same footnote is used with FILE_RECORDS). All three part-time required keywords covered in the table also are included in the list of optional keywords.

***3 Just for the record, since Jan 2007, I have changed the file model and its sundry versions at least 4 times.

The PSDD and specification show RECORD_TYPE and FILE_RECORDS as required. I don't see why FILE_RECORDS is required and FILE_NAME, LABEL_RECORDS, RECORD_BYTES are not.

NOTE: In the specification, IMPLICIT_FILE makes RECORD_TYPE, FILE_RECORDS, and RECORD_BYTES required, and IMPLICIT_FILE_ATTACHED adds LABEL_RECORDS to the list. The distinction between IMPLICIT_FILE and IMPLICIT_FILE_ATTACHED is clear and appropriate. However, neither of these object descriptions includes the optional elements for the File object, nor are these two objects listed as sub-objects of File.

Optional keywords in PSDD, but not in StdRef
PSDD

Optional keywords in StdRef, but not in PSDD
DESCRIPTION
ENCODING_TYPE
INTERCHANGE_FORMAT
REQUIRED_STORAGE_BYTES
UNCOMPRESSED_FILE_NAME

***3 I will wait on the resolution of these issues before the model is updated.

********* More on File Objects **************************

I couldn't help myself! I transposed the table in the standards reference in an attempt to identify how many file objects we really need. This was an interesting exercise, but we may not be able (or wish) to incorporate the results in the specification.

Ignore for the moment minimal detached labels and tape archives.

It turns out we need (at least) four types of File Objects (call them A,B,C,D).

Type A: Supports objects which have RECORD_TYPE = STREAM and use detached labels.
Type B: Supports objects which have RECORD_TYPE = STREAM and use attached labels.
Type C: Supports objects which have RECORD_TYPE = either FIXED_LENGTH or VARIABLE_LENGTH and use detached labels.
Type D: Supports objects which have RECORD_TYPE = either FIXED_LENGTH or VARIABLE_LENGTH and use attached labels.

The respective required keywords are:

Type	Required Keywords
A	RECORD_TYPE
B	RECORD_TYPE RECORD_BYTES
C	RECORD_TYPE RECORD_BYTES FILE_RECORDS
D	RECORD_TYPE RECORD_BYTES FILE_RECORDS LABEL_RECORDS

NOTE: The usage of RECORD_BYTES is non-standard for Type B and for VARIABLE_LENGTH Type C & D
(hence the 'at least' parenthetical comment above).

Minimal detached labels and tape archives require the addition of FILE_NAME, so we get two more types, Am and Cm which would be sub-objects of types A & C respectively.

Type	Required Keywords
Am	RECORD_TYPE FILE_NAME
Cm	RECORD_TYPE RECORD_BYTES FILE_RECORDS FILE_NAME

****
***

===== GAZETTEER_COLUMN

ok.

===== GAZETTEER_TABLE

has_Gazetteer_Column - cardinality should be 20, not 1..20

Good call. Fixed.

Under optional keywords, the StdRef lists "any". Is that even legal for a Specific Object?

***3 - Not that I remember.

===== HEADER

The specification matches the PSDD.

Optional keywords in PSDD, but not in StdRef
PSDD

===== HISTOGRAM

The specification matches the PSDD.

Optional keywords in PSDD, but not in StdRef
PSDD

===== HISTORY

The specification does not match the PSDD directly. The PSDD shows an optional keyword, PSDD, for HISTORY which is not stated explicitly in the specification. However, in the way that we use GAZETTEER_COLUMN within GAZETTEER_TABLE, the specification uses History_Group within HISTORY and History_Group does show the PSDD keyword, although it indicates that it is required not optional (see the next entry).

Optional keywords in PSDD, but not in StdRef
PSDD

===== HISTORY_GROUP

This is not in the PSDD or the StdRef. It only lists
one attribute, PSDD. PSDD is shown with a cardinality of
1..*; it should be 0..*.

I think I'm missing something here. This seems an
unnecessary artifice (unlike the GAZETTEER_COLUMN).

***3 Yes, this was my attempt to model something that the Stds Ref requires, the use of a Group for each History entry. This is unique in that where PSDD in other classes allow the simple inclusion of the keyword, History requires that the keywords to be nested within a Group.

Should
we use it to show the History Entry Elements? (i.e.,
VERSION_DATE
DATE_TIME
NODE_NAME
USER_NAME
SOFTWARE_DESC
USER_NOTE)

===== IMAGE

The specification matches the PSDD.

Optional keywords in PSDD, but not in StdRef
LINE_DISPLAY_DIRECTION,
SAMPLE_DISPLAY_DIRECTION
PSDD

***3 Stuff to consider.

===== Index_Table

Association - has_Index_Columns, but the specification

does not contain an Index_Column object analogous to Gazetteer_Column

Changed name to be consistent with Gazetteer.

has_Index_Columns should have cardinality of 7..8 with a caveat that any given Index_Column can be replaced by a corresponding attribute if it is single valued (I don't know how to do that in the specification).

should also have Association - has_Columns with cardinality 0..*

Nice, I should have thought of that. Change made.

===== IMAGE_MAP_PROJECTION

This feels like a misplaced Data Object - see Anne's comments in her section on Data Set Map Projection.

***1 Yes, it needs more discussion. I mentioned to Anne that it seems that IMAGE_MAP_PROJECTION is a set of parameters and so is a description of a conceptual thing. It does not describe a sequence of bits.

***3 Inconsistencies that need resolution.

Once again there are issues with required vs. optional elements between the StdRef, the PSDD, and the specification.

There are four elements which the PSDD lists as optional, the StdRef lists as required and the specification shows with cardinality of 1 (matches the StdRef not the PSDD):

FIRST_STANDARD_PARALLEL,
REFERENCE_LATITUDE,
REFERENCE_LONGITUDE,
SECOND_STANDARD_PARALLEL

There are an additional four elements that both source documents show as option which are missing from the specification:

DATA_SET_ID,
HORIZONTAL_FRAMELET_OFFSET,
IMAGE_ID,
VERTICAL_FRAMELET_OFFSET

2. PDS3 Specification - A. Raugh Review 1 - Response 3

Hi all,

This resend includes the latest comments on Anne's cross checking review.

080501 - Review of responses from Data_Set_Mission to End - WG
080417 - Review of responses from beginning to Data_Set_Mission - WG
080414 - Response to review comments - Steve
080414 - Review comments - Anne

thanks,
steve

Review comments
---

1) Changed Housekeeping to Data_Set_Housekeeping.

2) When class and or attributes are included in the model from the catalog database, that are not included in the standards document, flag them as such.

3) Qube tabled until formal review.

>080501

4) List the subsummed subobjects at the beginning of each relevant section of the document. (e.g. data_set_mission). Also add a note explaining the logic used to determine the resultant cardinality. (i.e. the PDS use of N/A and the resultant cardinality = 1.

5) Add a new column to the document to indicate whether the attribute/association is Required, Optional or Not Allowed.

6) Consider removing "uses_pointer" from the data description classes such as Table and whether it is sufficient to have the "pointer" modeled in the "Tagged Data Object"

classes.

7) Add a abstract class named "Repeating_Group" for classes like "software_online"

8) Model additional pointer types such as "filename", description, note, and text.

9) Add a note to describe the differences between generic objects (allowed optional keywords) and specific object (required keywords only) and how these concepts are reflected in the document.

10) Determine the reason for having two "has_file_*" associations and why one generic will not do. Generalize if there is no valid reason.

11) Add a note describing the use of PSDD and what it means.

12) A single Document pointer can have multiple values. (i.e. multiple files)

13) Add the following anomalies:

- Data_Supplier is a generic catalog object.
- Directory is a generic catalog object.
- The "implicit" relation between data_set and data_product is obvious, might exist in the form of certain index_tables, and can be calculated from the relationship between product and data set. Is this sufficient?
- Description pointer - If the thing pointed to has its own label then there is an implicit undocumented relationship between the two things. Does this need to be explicit.
- Multi-valued pointers. Allowed?

At 07:14 PM 4/14/2008, J. Steven Hughes wrote:

Hi Anne,

Responses below.

I have denoted the responses with a ***1 for an open issue and ***2 for a proposed PDS3 anomaly.

thanks,
steve

At 09:02 AM 4/14/2008, Anne Raugh wrote:

Steve,

I spent the morning checking and cross-referencing my assigned objects from the pdsdd.full file you sent. Notes (frequently accompanied by or in the form of question) are below, with general questions/comments at the very end. I hope they all make some sort of sense...

-Anne.

==========

Files Referenced:

1. PDS Information Model Specification (HTML format) Version 0.070916m
2. Standards Reference PDF files as available from the PDS home page
3. pdsdd.full file 1r69
4. PSDD PDF file as available from the PDS home page

Specific discrepancies found between sources are noted below.

DATA_SET_INFORMATION

This object has been absorbed into the Data_Set class along with the other DATA_SET_* sub-objects.

Yes.

DATA_OBJECT_TYPE has the wrong cardinality - should be 1..*

***2 If we are modeling the as-is implementation of the PDS3 standards, then the cardinality is probably 1. DATA_OBJECT_TYPE is used in a specific object that does not repeat. It is also used to load a table that allows only one value. In any case this is a long standing anomaly.

ARCHIVE_STATUS should have a cardinality of 0..1. (It is allegedly being phased out and is currently optional. This in itself is an anomaly, as DATA_SET_INFORMATION is a specific object.)

Fixed to match DD.

In the Data_Set class description:
- What is "Data_Set_Organization" in that association?
- What is "Data_Set_Collection_a"?

Deleted. They were part of an early attempt to separate the physical and logical descriptions of a data set.

- How are the curated_by and distributed_by associations established?

A housekeeping template called inventory node media is used to associate a data set to all nodes from which it is distributed. The EN data engineer is responsible for completing this template and establishing the curated_by node.

Note that it is NOT true that the NODE object contains a DATA_SET_ID.

Yes, the inventory node media template is used to provide the data set ids.

- How is the has_Resource association established?

***2 Currently this is done as part of housekeeping. The nodes provide resource_information templates and they are linked by the DE to the data sets.

>>>080417

DATA_SET_MAP_PROJECTION, DATA_SET_MAP_PROJECTION_INFO & DS_MAP_PROJECTION_REF_INFO

The Data_Set_Map_Projection class is missing an attribute corresponding to DATA_SET_ID - the only element actually in the DATA_SET_MAP_PROJECTION object (I don't know the cardinality allowed).

Fixed. Since it is a specific object and not in a repeating group, the cardinality is 1.

It seems to contain all
the elements of the DATA_SET_MAP_PROJECTION_INFO object, but is missing
the REFERENCE_KEY_ID element from the DS_MAP_PROJECTION_REF_INFO sub-
object of the *_INFO object. This would have a cardinality of 1..*.

Fixed.

Presumably the missing DATA_SET_ID attribute is why there is no
association with the Data_Set class listed. Wouldn't the required
REFERENCE_KEY_ID attribute also lead to an association with a
reference class?

Good question. I added the explicit associations. The data_set_id and reference_key_id attributes are the means (foreign keys) by which the PDS implements the associations.

Note that the IMAGE_MAP_PROJECTION, a data object, requires the
DATA_SET_MAP_PROJECTION, a catalog object, be included as a sub-object
of itself. I know of no other case in which a catalog object is
subordinate to a data object. In all other cases the relevant catalog
object is associated by requiring an element that contains a key
field, like MISSION_NAME or DATA_SET_ID. This may be because the
DATA_SET_MAP_PROJECTION object does not contain a unique identifier
of its own. DATA_SET_ID appears to be the identifier, but it is not
clear from the description that the DATA_SET_ID value in the
DATA_SET_MAP_PROJECTION object must be single-valued.

***2 Good call. This should be an association given the way it is modeled in the specification. A relational (ODL) implementation should use a foreign key, the unique identifier of the map projection.

[Actually, Appendix B of the SR claims that IMAGE_MAP_PROJECTION is
a catalog object, but a) it is not a specific object; b) it is
listed as an optional sub-object of FILE - a list which, apart from
IMAGE_MAP_PROJECTION, consists entirely of data objects; and c) the

standard value set for DATA_OBJECT_TYPE includes the entry "IMAGE_MAP_PROJECTION".]

***1 It seems that IMAGE_MAP_PROJECTION is a set of parameters and so is a description of a conceptual thing. It does not describe a sequence of bits. A FILE and its other "subobjects" do describe sequences of bits.

DATA_SET_MISSION

This is subsumed in the Data_Set class. The one keyword appears with a cardinality of 1..*, but I believe relatively recent tech discussions concluded that the cardinality may, in fact, be 0..*. I can't find an an SCR for it, though. The object is certainly required, so if a value of "N/A" satisfies the 1..* requirement, then it's OK.

Will assume that N/A satisfies the 1..* requirements. Again, the subsumption is accomplished first by adding the explicit association. (I.e. has_mission). The inclusion of the foreign key (i.e. Mission_name) as an attribute addresses the current PDS implementation. The cardinalities should match.

DATA_SET_REFERENCE_INFORMATION

This is subsumed in the Data_Set class. The one keyword appears with a cardinality of 1..*, which is OK provided "N/A" is acceptable as a value.

Agree

DATA_SET_RELEASE

I think it is true that the cardinality is not actually restricted to 1 for the DATA_SET_ID, but I'm not sure as the object is not documented elsewhere.

Since this object is used to load a table, its cardinality is 1.

Why does this not have an association with at least the Data_Set class, and/or conversely?

Good call. An association with data set was added. (part_of_data_set) The inverse was never implemented since a single release can not be part of more than one data set.

I cannot find documentation of this object in any of the SR or PSDD appendices.

DATA_SET_TARGET

This is subsumed in the Data_Set class. The one keyword appears with a cardinality of 1..*, which is OK provided "N/A" is acceptable as a value.

Agree

DATA_SUPPLIER

Unlike nearly all other catalog objects, DATA_SUPPLIER is a generic object, not a specific one. It does have optional elements.

***2 Agreed, and this is allowed since the object was not designed to load a catalog database. This raises a question that will have to be answered in the design phase, after we finish this task. Optional elements and especially elements with cardinalities greater than one have added baggage when a traditional catalog (e.g. relational database) is to be implemented. This has limited us in the past (e.g. data_object_type). It would be nice to consider new implementation options.

DIRECTORY

Why is the second association "has_Directory_File" and not "has_File"?

to distinguish it from the has_File used in the Volume class. Actually I was not consistent and should have named this association has_File_Directory, similar to that used in similar situations. I have changed the name but am open to an alternate naming scheme.

Note that the DIRECTORY object is a generic object, not a specific one. This is inconsistent with classing it as a "catalog object". It also doesn't make sense to me that, in the IM Spec, Directory is a catalog class while File is a data class.

***1 Regarding the last point, File describes a sequence of bits. Directory is describing an organization. (Granted under UNIX a directory is a file however it is the generic case we are modeling.) I agree that it is not exactly a catalog object. Needs more discussion.

DISCIPLINE_DESCRIPTION

This specific object is not a sub-object of any other PDS object and is not listed in the Information Model Spec. Neither is there a reference to it in the SR or PSDD appendices. That is, it exists only in the pdsdd.full file. I wonder if this is a vestige of some catalog object long ago revised or deleted?

Good call. I must have entered this one while half asleep.

It is a very real catalog object designed for V1.0 and that survived with no changes. I have added the class and an association (has_discipline) from Node to Disciplne_Discription. The class name could be cleaned up a little.

DOCUMENT

I'm not entirely sure it is true that ENCODING_TYPE will be either zero or one. Since it is possible to have more than one file constitute a document, there is no clear constraint on the number of potential encoding types for a single DOCUMENT.

Made cardinality "multiple".

Also, it is frequently true that in "data" pointers for DOCUMENT objects there is more than one file listed in a single pointer. This is inconsistent with the attributes of the Data_Object_Pointer class in section 9.2 of the IM Spec. SR Chapter 14 ("Pointer Usage") doesn't specifically prohibit this, although it doesn't show any examples of multi-file pointers, but it's a DOCUMENT convention that's been used for years and is included in the sample DOCUMENT object in SR Appendix A. I have also seen cases where sometimes all the files of the same encoding type are grouped together (all the ASCII files, all the PDF files, etc.), and others where all the logical components of a single document are grouped together (the ASCII text and JPEG graphics, e.g.).

***1 Needs more discussion.

Why does the Document class have "File" in its "Referenced from" list when other top-level objects (like Time Series) don't? (Most seem to, though - like TABLE and QUBE. Perhaps an oversight in the other object(s)?)

This is the inherited association issue that has a TBD fix.

ELEMENT

START_BYTE is optional in an ELEMENT object and thus should have a cardinality of 0..1.

Fixed.

EVENT, EVENT_INFORMATION & EVENT_POSITIONAL

None of these are in the IM Spec. Neither could I find any mention of them in the SR and PSDD appendices. They exist only in the pdsdd.full file.

They were designed for V1.0 and seldom if ever used. The catalog streamlining effort seemed to provide the coup de grace. If the team thinks we need to model them I will,

otherwise we can just talk to them.

These appear to be descriptive keyword collections. The descriptions state they are catalog objects, but they have only generic object definitions, not specific ones as would be expected for catalog objects (although they have no optional keywords or sub-objects). Are these, perhaps, SPICE-related?

Additional Notes
==========

If we're going to use "PSDD" in the attributes listings, we're going to have to define what it means somewhere - section 6 or 5, most likely.

***1 Yes I agree. Needs discussion.

In tracing back through the class hierarchy for Data_Set, I get to Catalog_Description - but the "Referenced from" links don't work. Should they even be there?

It is not at all clear to me how the "Associations" for any particular class were deduced. In the Data_Set class, for example, most of the has_* relations are clearly defined by keyword values in the DATA_SET object.

Yes, if a keyword is a primary key for one class and then is found in another class the assumption made is that it is a foreign key and represents a relation. These are confirmed by the joins in the catalog applications. Of course they are not that accessible.

But there are three glaring exceptions: has_Data_Set_Organization

This was an early attempt to make the organization more explicit. As I mentioned it has been deleted.

has_Product_Implicit,

***2 This is an anomaly and needs more discussion. There is an explicit association from data product to data set through the use of the foreign key data_set_id in the Identification_Data_Elements class. It could be argued that there is an explicit relationship from data set to data product in an index_table, but I have to admit that it is a stretch. This was an attempt to model something that seemed obvious.

and has_Resource.

***2 A keyword should exist. It is a housekeeping object and the DE handles the anomaly.

I have no idea what the first
is supposed to represent, and trying to follow the HTML links takes you
nowhere. The second seems to be a backward relation from the data product,
which will have a DATA_SET_ID keyword, but I don't see a justification for
elevating this to the level of association while other backward-relations
are listed merely as "Referenced from". And the third one seems to be
a back-reference from a Resource class, which doesn't have a corresponding
object in the PSDD or SR. (The closest is RESOURCE_INFORMATION, which is a
subset of the DATA_SET_HOUSEKEEPING object, which in turn exists only in
the pdsdd.full file.

Changed name from Resource to Resource_Information.

There is a HouseKeeping class in the IM Spec, and
the associations involved with it are also problematic, logically.)

I note that the only pointer type included in the IM Spec is the data
object pointer. While I'm happy to ignore structure pointers as a
syntactical implementation detail,

I agree.

the case for description pointers
is not so clear. If the thing pointed to by the description pointer
has its own label, then there is an association here between products
that we've got to document somehow. Even if there isn't another label

involved, pointing to an additional file external to both the label
and the data in order to provide additional meta-data is a relationship
that probably needs to be documented somehow. So I'm thinking maybe
one additional pointer class needs to be defined, but I'm not sure you
can document its potential associations without pouring cement into the
already very muddy waters.

***2 We might want to model the simple case and leave the others as anomalies.
Anyway this seems to be a discussion item.

3. PDS3 Specification - R. Joyner Review 1 - Response 2

Hi all,

This resend includes the final comments on Ron's cross checking review.

Certain responses are denoted with a ***1 for an open issue or for further discussion,
***2 for a proposed PDS3 anomaly, and ***3 for inconsistencies in either the PSDD or
the Stds Ref.

thanks,
steve

080508 - Review of responses - WG
080501 - Response to review comments - Steve
080430 - Review comments - Ron

Final Review comments
---
1) The case of BIT_COLUMNS/COLUMNS with ITEMs and the use of BITS/BYTES
was added as an anomaly to be addressed in PDS4.

>Review Comments / Issues:
>
>Miscellaneous comments:
>
> (1) Change: Data_Object_Description_wP
> to: Data_Object_Description_wPtr

The Pointer relationship will be removed from this class and its name changed.

>
>
> (2) - can you make the font smaller in the html version

Will be updating the HTML in the next version or two.

>
> (3) - when you click on a keyword (e.g., alias_name), there isn't a definition
> - why not instead point to the online data dictionary
> http://pds/tools/data_dictionary_lookup.cfm
> - wouldn't this save you a whole lot of work
> - I have only verified a few of the definitions provided in the spec against those in
> the ddict -- some match and some do not match

The data dictionary is downloaded and imported into the tool. The data element definitions should match. Object definitions are not loaded from the data dictionary but pasted in.

>
>(4) The Standards Reference captures a number of examples for each OBJECT.
> Is this something that goes into the DUMMIES version of the model ??
>

Good idea.

>
>
>Object comments:
>
> (1) NAME = ALIAS
> - no comments
>
> (2) NAME = ARRAY
> - the class description that is provided is a summary of the full
> description provided currently in the Standards

The plan is to update the class descriptions in a later phase.

>
> Is it approprate to have a field like abstract_text where you would
> put the other information that is listed in the Standards Ref
>
> Example:
>
> The ARRAY object is provided to describe dimensioned arrays of homogeneous objects. Note

> that an ARRAY may contain only a single sub-object, which can itself be another ARRAY or
> COLLECTION if required. A maximum of 6 axes is allowed in an ARRAY. By default, the
> rightmost axis is the fastest varying axis.
>
> The optional "AXIS_*" elements are used to describe the variation between successive objects
> in the ARRAY. Values for AXIS_ITEMS and "AXIS_*" elements for multidimensional arrays
> are listed in axis order. The optional START_BYTE data element provides the starting location
> relative to an enclosing object. If a START_BYTE is not specified, a value of 1 is assumed.
>
> - change: name_
> to: name

Fixed. The one change will take care of the rest metioned below.

>
> - why does referenced_from include: array

An array can contain an array.

>
>
> (3) NAME = BIT_COLUMNd
> - the class description that is provided is a summary of the full
> description provided currently in the Standards Ref
>
> Is it approprate to have a field like abstract_text where you would
> put this ancillary information
>
> - change: name_
> to: name
>
> - bits is required for BIT_COLUMNs without ITEMS; but optional for BIT_COLUMNs with items
> - how do you model that one ???

***2 Anomaly Bit_Column needs to be subclassed so that there are two different classes.
***3 To Elizabeth - BITS are optional in Std Ref.

>
>

> (4) NAME = BIT_ELEMENT
> - the class_description doesn't match that found in the ddict.
>
> PDS Spec: The BIT_ELEMENT object provides a means of defining a
> lowest-level component of a data object.
>
> ddict: The bit_element object identifies a bit string embedded
> in a element.
>
>
> (5) NAME = CATALOG
> - the class description provided is the full description found in the
> Standards Ref. (which doesn't match the convention used in the Data Object
Descriptions)
>
> - the CATALOG object is listed as a GENERIC object (which would indicate that
> the object can contain any keyword in the PSDD). But, I think in practise
> that the CATALOG object should not have PSDD as an attribute.

***2 PSDD is currently considered an anomaly and will be readdressed in PDS4.

>
> You also have the issue of the CATALOG object containing all of these
> CATALOG pointers (that aren't in the ddict):
>
> ^MISSION_CATALOG = "MISSION.CAT"
> ^INSTRUMENT_HOST_CATALOG = "INSTHOST.CAT"
> ^INSTRUMENT_CATALOG = "INST.CAT"
> ^DATA_SET_COLLECTION_CATALOG = "DSCOLL.CAT"
> ^DATA_SET_CATALOG = "DATASET.CAT"
> ^REFERENCE_CATALOG = "REF.CAT"
> ^PERSONNEL_CATALOG = "PERSON.CAT"

***2 Catalog pointers are an anomaly.

>
> - change: logical_volume_path_name
> to: logical_volume_pathname

logical_volume_path_name is what is in the PSDD.

>
> (5b) NAME = CATALOG_DESCRIPTION
> - I assume this is the parent class for the set of CATALOG thingys
> - change the class description to indicate that this is the
> parent entity

Done

> - also same for Archive_Collection and Collection2
> - these have been removed per Steve (20080417)

Yes

>
>
>
> (6) NAME = COLLECTION
> - the supplied class description doesn't match that found in the Standards Ref
>
>
> (7) NAME = COLUMN
> - the class description that is provided is a summary of the full
> description provided currently in the Standards
>
> Is it approprate to have a field like abstract_text where you would
> put the other information that is listed in the Standards Ref
>
> - change: name_
> to: name
>
> - bytes is required for COLUMNs without ITEMs
> - bytes is optional for COLUMNs with ITEMs
> - how to model that ???
>

***2 Probably by subclassing Columns.

> - Referenced from: add INDEX_TABLE (required)
> add SERIES (required)
> add SPECTRUM (optional)
> add GAZETTEER_TABLE (required)
> - rename GAZETTEER_COLUMN to GAZETTEER_TABLE

Referenced_by has a TBD fix.

>
>
> (8) NAME = CONTAINER
> - change: name_
> to: name
>

> - Referenced from: add INDEX_TABLE
> add SERIES
> add SPECTRUM
> add GAZETTEER_TABLE
> - rename GAZETTEER_COLUMN to GAZETTEER_TABLE

Referenced_by has a TBD fix.

>
> - Can CONTAINER be a sub-object of SPREADSHEET and repeat FIELD objects ???

Not as currently modeled in the PSDD.

>
>
> (9) NAME = DATA_PRODUCER
> - the supplied class description doesn't match that found in the Standards Ref
>
>
>(10) NAME = DATA_SET
> - the supplied class description doesn't match that found in the Standards Ref
>
> - change: data_set_collection_member_flag
> to: data_set_collection_member_flg

Good call, fixed.

>
> - mission_name === in has_Mission class
> - instrument_host_id === in has_Host class
> - instrument_id === in has_Instrument class
> - target_name === in has_target class
> - reference_key_id === in has_reference class
>
> - not sure I understand has_Data_Set_Organization
>

It has been deleted

> - not sure I understand has_Product_Implicit

***2 Index tables are the closest we have to an implementation for this relation. It is an anomaly

>

> - referenced from: archive_collection (link doesn't work)

deleted

> - referenced from:
>
>(11) NAME = DATA_SET_COLL_ASSOC_DATA_SETS
> - removed

>
>(12) NAME = DATA_SET_COLLECTION
> - the supplied class description doesn't match that found in the Standards Ref
>
> - reference_key_id === in has_reference class
>
>(13) NAME = DATA_SET_COLLECTION_INFO
> - removed
>
>(14) NAME = DATA_SET_COLLECTION_REF_INFO
> - removed
>
>(15) NAME = DATA_SET_HOST
> - removed
>
>(16) NAME = DATA_SET_HOUSEKEEPING
> - removed
>
>
>

4. PDS3 Specification - M. Gordon Review 2 - Response 2

Hi all,

This resend includes the final comments on Mitch's 2nd cross checking review.

Certain responses are denoted with a ***1 for an open issue or for further discussion, ***2 for a proposed PDS3 anomaly, and ***3 for inconsistencies in either the PSDD or the Stds Ref.

thanks,
steve

080508 - Review of responses - WG
080501 - Minor update regarding volume - Mitch
080429 - Response to review comments - Steve
080428 - Review comments - Mitch

Final Review comments
---
1) ***2 TARGET_REFERENCE_INFORMATION is an optional object of TARGET. Specific objects should not allow optional objects.

At 02:29 PM 4/28/2008, Mitch Gordon wrote:
Hi Steve,

Here are my comments on the second set of objects.

Cheers,

Mitch
==========

Files Referenced:

1. PDS Information Model Specification (HTML format) Version 0.070916m
2. Standards Reference 3.7, Mar 20, 2006
3. PSDD: pdsdd.full file 1r69

==========

Again I've changed the order to parallel the order in
the specification.

==== TARGET

- There are no inconsistencies between the PSDD & StdRef!

- TARGET has one required object, TARGET_INFORMATION, and
  one optional object, TARGET_REFERENCE_INFORMATION. Our
  standard approach has been applied here: not listing
  these 'embedded' objects directly, but capturing all of
  their elements directly under TARGET. The tricky bit is
  that REFERENCE_KEY_ID is a required element of
  TARGET_REFERENCE_INFORMATION which is an optional object
  of TARGET. I think this means the cardinality of
  REFERENCE_KEY_ID should be 0..*, not 1..*.

Yes, this has been fixed.

- TARGET shows TARGET_ID with a cardinality of 1, but
  TARGET_ID does not appear under target in either the
  PSDD or the StdRef.

Target_Id has been removed. There is no such element.

==== TARGET_INFORMATION

- Not listed directly in the specification: subsumed into
  TARGET with all keywords retained.

==== TARGET_REFERENCE_INFORMATION

- Not listed directly in the specification: subsumed into
  TARGET with all keywords retained.

==== VOLUME

- NOTE: The StdRef lists Volume as a Data Object, we have
  it listed as a Catalog Class in the Specification.
  We're probably right, but that makes this an anomaly.

***2 Will add it as an anomaly.

>080501 - Volume - "has_file" the cardinality should be 1..*

***1 I agree with Anne that the subobject is not required. Volume is a generic object.

- I think "has_Volume_I" should have cardinality of 1..*

Yes, fixed.

- Product_Type should have cardinality of 0..*

Fixed.

- Optional keywords in dd.full, but not in StdRef:
  PSDD

***3 to Elizabeth.

==== SPICE_KERNEL

- Looks okay.

==== SPREADSHEET

- Looks okay.

==== TABLE

- The PSDD lists an optional element, PSDD, which is missing from both the StdRef and the specification.

***3 to Elizabeth - Has been added.

==== TEXT

- The PSDD lists an optional element, psdd, which is included in the specification, but is missing from the

***3 to Elizabeth.

StdRef.

5. PDS3 Specification - D. Simpson Review 3 - Response 2

Hi all,

This resend includes the final comments on Dick's 2nd cross checking review.

Certain responses are denoted with a ***1 for an open issue or for further discussion, ***2 for a proposed PDS3 anomaly, and ***3 for inconsistencies in either the PSDD or the Stds Ref.

thanks,
steve

080508 - Review of responses - WG
080502 - Response to review comments - Steve
080502 - Review comments - Dick

thanks,
steve

Final Review comments
---
1) SOFTWARE will be moved to the Catalog Description section.

2) ***2 SOFTWARE is an anomaly. It is currently used as a catalog object. However, shouldn't it describe digital data?

3) Move RESOURCE_INFORMATION to Operational_Description since it is a component of DATA_SET_HOUSEKEEPING.

4) In the spec document, any inherited attribute should be classified as inherited. Also indicate if that attribute was restricted in any way.

At 10:36 AM 5/2/2008, Dick Simpson 650-723-3525 wrote:
Steve:

Review of second set of classes/objects

Dick

- -

Files referenced:
1. PDS3 Spec 0.070916m

2. PDS SR 3.7
3. pdsdd.full 1r69

REFERENCE:

PDS SR shows REFERENCE being called from DS_MAP_PROJECTION_REF_INFO

***3 To Elizabeth

RESOURCE:

No RESOURCE-like object is in PDS SR

The class will be flagged as being implemented but not modeled in PDS3.

PDS3 Spec shows this as RESOURCE, pdsdd.full as RESOURCE_INFORMATION

***3 to Elizabeth

Attribute resource_link = URI (not URL?)

Good call. Fixed.

Why is STATUS_TYPE = PENDING in pdsdd.full?

***3 to Elizabeth.

Why is OBJECT_TYPE = GENERIC in pdsdd.full?

*** to Elizabeth. Should be specific.

SERIES:

PDS3 Spec shows table_storage_type as attribute; but it should be an inherited attribute from TABLE.

Disagree. Series inherited the attribute and then changed the cardinality. Since it touched it, it owns it.

PDS3 Spec shows Attribute PSDD = 0..*; PSDD is not listed in PDS SR, but is shown in pdsdd.full.

***3 to Elizabeth.

SOFTWARE:

PDS3 Spec shows SOFTWARE as a data_object_description, but DESCRIPTION says it's a CATALOG object, which is where PDS SR puts it.

We are considering SOFTWARE to be describing a sequence of bits which makes it more like a data_object_description. Catalog vs Data Description is an anomaly.

PDS SR says SOFTWARE has three sub-objects: SOFTWARE_INFORMATION, SOFTWARE_ONLINE, and SOFTWARE_PURPOSE. The first and third appear to have been subsumed into SOFTWARE in PDS3 Spec, but the second has not. Why?

Software_Information is a group of attributes that does not repeat. It can be subsummed.
Software Purpose has only one attribute but repeats. It can be subsummed.
Software_Online is a group of attributes and does repeats. It can not be subsummed.

Association: uses_pointer = None may not be right; but I don't know enough about SOFTWARE to know what would be better.

None is correct.

SOFTWARE_INFORMATION:

Subsumed into SOFTWARE.

SOFTWARE_ONLINE:

The required attributes ON_LINE_IDENTIFICATION and ON_LINE_NAME have STANDARD_VALUE_TYPE DYNAMIC and NONE, respectively. Since both are supposedly unique, the rather casual approach to standard values seems inconsistent. This is a PSDD problem, not an issue for PDS3 Spec.

***3 to Elizabeth

SOFTWARE_PURPOSE:

This specific object has a single required attribute with the same name -- potentially confusing, therefore undesirable.

Subsumed into SOFTWARE

Luckily.

SPECTRUM:

table_storage_type is listed as an attribute whereas it should be an inherited attribute.

It touched it therefore ...

PDS3 Spec shows Attribute PSDD = 0..*; PSDD is not listed in PDS SR, but is shown in pdsdd.full.

***3 To Elizabeth.

6. PDS Specification - M. Gordon Review 3 - Response 2 – Sections 9.1-9.10 Cross Check

Hi all,

The following contains the compiled responses to Mitch Gordon's cross checking review of Sections 9.1-9.10.

Steve

080522 - Review of responses - WG
080521 - Response to review comments - Steve
080520 - Review comments - Mitch

Review of responses - 080522
---
1) Rename Implicit_File to File_Implicit in all occurrences.

2) Add as an anomaly the requirement that label_revision_note is required within PDS catalog files. The label_revision_note is not an attribute of the class. What is it an attribute of?

3) Add History as a Tagged_Data_Object with a NULL description.

At 10:44 AM 5/20/2008, Mitch Gordon wrote:
Steve:

Review of Sections 9.1 through 9.10.

Welcome back.

Mitch

- -

Files referenced:
1. PDS3 Spec 0.070916n
2. Standards Ref 3.7

Section 8 & 9 consistency

Section 8 uses Implicit_File in various instances;
Section 9 uses File_Implicit.

9.1 Data_Object

I compared the Referenced_From list for section
9.1 Data_Object with the subclasses for section
8.9 DO_Parent_Classes.

Here are the DO_Parent_Classes that do not have tagged
counterparts in the Data_Object referenced from list:

alias
gazatteer_table
history
spectral_cube

I think I'm okay with not including alias and history.

We have a tagged_index_table but not a tagged_gazatteer_table,
why not?

Why don't we have a tagged_spectral_cube?

Added tagged_gazetteer_table and tagged_spectral_cube.

Here is the converse - tagged objects in the Data_Object
referenced from list which do not have corresponding
DO_Parent_Classes:

Tagged_File_Data
Tagged_File_Implicit_Attached
Tagged_File_Other
Tagged_Header_FITS
Tagged_Header_VICR
Tagged_Series

Tagged_Software
Tagged_Spectrum
Tagged_Time_Series

Tagged_File_Implicit_Attached does trace back to Section 8 (8.22 Implicit_File_Attached, a subclass of 8.21 Implicit_File).

I assume that File_Implicit_Attached, along with Spectrum, Series, & Time_Series are not listed in the section 8.9 subclasses because they are sub-subclasses

Yes.

I'm not sure that I understand why we suddenly have an increase in File objects (File_Data & File_Other) which don't have counterparts in Section 8 and are not listed as subclasses of the File object.

***1 The file objects are needed for 1) combined detached and 2) software and document. Good call on making them subclasses. Done.

I believe we agreed to drop the *_FITS & *_VICR objects.

Done.

I'm not sure about Tagged_Software - but that is one of Dick's sections.

***1 Needs discussion. The use case you provided was an example of tagged software.

9.2 Descriptive_Data_Elements

Line 2, typo, "Class Description:IIn ..." should be "Class Description: In..."

Fixed.

'Referenced from' includes Data_Product_Image_FITS, Data_Product_Image_VICR, & Data_Product_Table_FITS. I believe we elected to remove these.

Done.

9.3 IDE_Ancillary

Okay.

9.4 IDE_Earthbase

Okay.

9.5 IDE_Spacecraft

Okay.

9.6 Identification_Data_Elements

'Referenced from' includes Data_Product_Image_FITS, Data_Product_Image_VICR, & Data_Product_Table_FITS. I believe we elected to remove these.

Done.

9.7 Label_Standards_Identifiers

Attributes	card	Ind
dd_version_id	1	O
label_revision_note	1	
pds_version_id	1	PDS3 O

pds_version_id should be required.

Fixed.

label_revision_note is
only required for catalog labels (StdRef 3.7, section 5.3.1)

Fixed: Now optional for product labels.

***1 Note for later update when I have internet access. Uncertain of which catalog classes.

'Referenced from' includes Data_Product_Image_FITS, Data_Product_Image_VICR, & Data_Product_Table_FITS. I believe we elected to remove these.

Done.

9.8 Tagged_Array

I'm a bit confused. tagged_array has the association: has_Data_Object_Description, "required", cardinality = 1, value = Array.

Actually these will be "inherited associations" when I get the fix made.

I'm good with all of that, but why are
Inherited Attribute & Inherited Association both none?
Shouldn't tagged_array inherit the full set of attributes and associations that belong to Array?

No, Tagged_Array is not a subclass of Array. It is a class with an association to Array. More specifically it is composed of Array, a pointer, and a data object.

If so, we may not want to repeat them here. Would it be meaningful to use 'see Array' instead of 'none'. I think I would find it more clear.

9.9 Tagged_Collection

Inherited Association: has_Data_Object, cardinality =1, value = Data_Object, Ind = O. Here's the tricky bit, collection has four optional objects - no specific object is required, but a collection must have at least one of the optional objects (null collections are not allowed). Should Ind = 1 for has_Data_Object?

Yes, fixed.

***1 Should the four associations (e.g. has_Collection) be combined into one association with four options.

9.10 Tagged_Data_Object

I compared 8.9 DO_Parent_Classes with the section 9.10 Tagged_Data_Object subclasses.

Here are the DO_Parent_Classes that do not have tagged counterparts in the tagged_data_object subclasses list.

alias
gazatteer_table
history
index_table
spectral_cube

Are gazatteer_table and index_table missing because they are subclasses?

Added gazatteer_table. Index_Table exists.

Why is spectral_cube missing?

Added Tagged_spectral_cube. However the Spectral_cube still needs to be fleshed out. Has asked Elizabeth for the latest.

Here is the converse - subclasses of tagged_data_objects which do not have corresponding DO_Parent_Classes:

Tagged_File_Data
Tagged_File_Other
Tagged_Software

Same comments as section 9.1

Same fixes.

7. PDS Specification - A. Raugh Review 2 - Response 2 – Cross Check Sections 9.11-9.20

Hi all,

The following contains the compiled responses to Anne Raugh's cross checking review of Sections 9.11-9.20.

thanks,
Steve

080522 - Review of responses - WG
080521 - Response to review comments - Steve
080521 - Review comments - Anne

Review of responses - 080522
---
1) Document can have multiple-valued pointers.

2) Tagged_file_data, tagged_file_other fixed to have data objects.

3) Implicit file added as an anomaly.

>Review of Sections 9.11-9.20
>==========
>
>What does "Ind" stand for in the tables? The paragraph at the end of Section 6
>has not been updated to explain it.

Updated.

>

>In the intro paragraph, "association" should be "associate".

fixed.

>
>The second paragraph after the class hierarchy 1st sentence list doesn't make
>grammatical sense.

No change needed.

>
>Things I noticed while trying to follow the logic...
>
> In the IDE_* classes, why is there no connection to the Mission?

***1 Mission_Name was not in figure 5.2. Is mission_name required.

> In 9.7, the LABEL_REVISION_NOTE appears to be required. In fact, it is
>   only required in catalog files, not labels.

Fixed

> In 9.8, the "Class Description" is for tha parent class, not the Tagged_Array
>   class.

Yes, a wholesale cleanup of definitions is required.

> In 9.9, the "Class Description" doesn't actually refer to "tagging". Is it
>   the right description?

Fixed but the update is not much better. See Tagged_Array.

>
>9.11 Tagged_Document
>
>The "Class Description" paragraph doesn't describe this class.
>

It is the boiler plate description to be updated later.

>If the pointer is optional - and I don't agee that it is - then shouldn't the
>entry in the mysterious "Ind" column be "O"?

***1 I don't understand the problem since the pointer is not optional but required and allows more than one pointer.

>
>I can only think of one case where a data object pointer might be null - when
>there is no bit string to point to. If you allow this degenerate case, then
>the pointer is logically null and you might consider making it optional (as
>opposed to designating a null pointer value). In all cases where the data
>object exists, the tagged object must indicate where it is - that's the whole
>point of the class. So logically, the pointer should never be optional. The
>omission of a pointer element in a series of ODL statements is a syntactical
>convenience dependent on the specific PDS implementation. Since we're
>supposed to be implementation-agnostic, I maintain that the pointer is,
>logically, always required.
>
>Moving on...
>
>9.12 Tagged_File
>
>The "Class Description" paragraph doesn't describe this class.
>
>9.13 Tagged_File_Data
>
>The "Class Description" paragraph doesn't describe this class.
>
>I don't understand what this is supposed to represent, and there's not enough
>information following any of the links to figure it out. Most importantly,
>what is a "Combined_Data_Product"?

***1 This is the combined detached label class.

>
>How is it possible that this class can require a data object description and
>a pointer to that object, but NOT require the data object itself?
>

fixed.

>9.14 Tagged_File_Implicit
>
>The "Class Description" paragraph doesn't describe this class.
>
>I'm not sure where this class is trying to go. The hierarchy presents a
>problem to me, because the implicit file doesn't seem to be to belong in
>the data_object hierarchy at all. If anything, it should be at a higher
>level, because it is the implicit file that "contains" all the tagged
>data objects that might appear in the file. And when those tagged data
>objects are in different files, the implicit file object becomes a very
>nebulous beast, with no meaningful attributes.

***1 The implicit file class is in concept equivalent to the explicit file class. It simply describes a sequence of bits that has file boundaries, the data file. The existence of data objects (e.g. image) that to map to a subset of that sequence of bits is a redefinition or overlay on those bits. This is similar to the concept of union structures in programming languages, redefining the same memory. There is no hierarchy or containment. I.e. In the PDS, a File is not defined as consisting of anything as Table consists of columns.

>
>So I'm thinking if the tagged_implicit_file has associations, it is only
>with tagged data objects. We can argue about whether it has a true
>data object description, but it doesn't have a pointer of its own, ever,
>and I'm not convinced it has a data object of its own, either.

Data Object Descriptions - A description is one or more attributes describing a sequence of bits. Record_Type alone seems sufficient.

Pointer - I agree that the pointer does not exist. fixed.

Data Object - The sequence of bits with file boundaries.

>
>I'm thinking that all the attributes assigned to the implicit_file object
>are, in fact, attributes of the data object descriptions, but the
>actualy implementation treats them in a degenerate fashion that losses some
>information.

Record_Type, File_records, and record_byte seem to be attributes of a sequence of bits interpreted as a file with file boundaries.

>
>9.15 Tagged_File_Implicit_Attached
>
>The "Class Description" paragraph doesn't describe this class.
>
>All the problems with the higher level class reappear here, so addressing
>the first should pretty much take care of the second.
>

Agree.

>9.16 Tagged_File_Other
>
>The "Class Description" paragraph doesn't describe this class.
>
>I don't understand the point of this class. I also don't understand the

>"has_labeled_explicit_file_object" associations in the Document_product and
>Software_Product classes - because they don't have explicit file objects in
>their labels.

Product has two subclasses, data product and combined detached data products. The later has a required explicit file object. In turn the explicit file object has the tagged document, software, etc. object.

>
>9.17 Tagged_Header
>
>The "Class Description" paragraph doesn't describe this class.
>
>I thought we had expunged all subclasses based on the value of an attribute,
>in which cans the two subclasses here need to go away. There is no difference
>between header objects for VICAR headers vs those for FITS headers.

done

>
>9.18 Tagged_Histogram
>
>The "Class Description" paragraph doesn't describe this class.
>
>The "Data_Product_Image_VICR" class included in the "Referenced from" list
>doesn't exist (nor should it).
>

Removed.

>9.19 Tagged_Image
>
>The "Class Description" paragraph doesn't describe this class.
>
>"Data_Product_Image_VICR" is referenced here as well.
>

Removed

>9.20 Tagged_Index_Table
>
>The "Class Description" paragraph doesn't describe this class.
>
>I thought Index_Table was going to be treated as a separate class, no? Why
>is it showing up here as a subclass of Table?

Fixed.

8. PDS Specification - D. Simpson Review 4 - Response 2 - Sections 9.21-9.30 Cross Check

Hi all,

The following contains the compiled responses to Dick Simpsons's cross checking review of Sections 9.21-9.30.

Steve

080522 - Review of responses - WG
080520 - Response to review comments - Steve
080519 - Review comments - Dick

Review of responses - 080522
---
1) Remove has_Secondary_TDO association from Data Product

2) Review Spice_Kernel model with Boris. Specifically confirm whether a pointer exists.

3) Add Software product as an anomaly. It has no pointer.

4) Add Spectrum data product.

5) Remove Tagged_Time_Series.

6) Remove Attached_Detached distinction in model. It is an implementation issue.

At 09:18 AM 5/19/2008, Dick Simpson 650-723-3525 wrote:
Steve:

Review of Sections 9.21 through 9.30. I'm either losing my confidence
that this job can be completed or losing my understanding of the model.

Welcome back.

Dick

- -

Files referenced:
1. PDS3 Spec 0.070916n

9.21 TAGGED_PALETTE

Has_Data_Object and has_Pointer are not inherited because they are required whereas they are optional in the parent class. Why isn't has_Data_Object_Description inherited? It's required in both.

I think that you understand the model more than you take credit for. The problem is probably that I fixed the code so that cardinality = 0 indicates an Optional attribute or association with Cardinality = 1. However I did not fix the code to show inherited attributes if they had been changed locally.

The association has_Data_Object_Description has a value change so is in the same category of the other two associations.

The "inherited" change should be in the next version.

However, the effective results are the same. A tagged Palette is required to have the three associations each with one value.

TAGGED_PALLETTE is referenced from four things; why is it "secondary" in each case, and why is it has_Secondary_Data_Description under Tagged_File_Data but has_Secondary_TDO under the other three?

***1 That is the way it was modeled. More discussion needed since at one point there was a question in the WG as to why we still need to identify secondary tagged objects.

Why is it referenced from two kinds of labels, one data product, and Tsgged_File_Data? I'm losing my sense of hierarchy ...

Exactly, the PDS data model is not a hierarchy. It is a network. Once defined, something can be referenced from where ever it makes sense.

9.22 TAGGED_QUBE

Same as TAGGED_PALETTE except there is a fifth referenced_from (Data_Product_Qube).

However, the effective results are the same. A tagged QUBE is required to have the three associations each with one value.

9.23 TAGGED_SPICE_KERNEL

Same questions as TAGGED_PALETTE

I'm puzzled that has_Pointer could be optional or have cardinality 0.

SPICE-KERNEL has no pointer.

9.24 TAGGED_SERIES

Same questions as TAGGED_PALETTE except that the Association versus Inherited Association categorization is exactly backwards.

However, the effective results are the same. A tagged Series is required to have the three associations each with one value.

The changes were not made in Series. They were made in Table and inherited.

9.25 TAGGED_SOFTWARE

Same questions as TAGGED_SPICE_KERNEL

However, the effective results are the same. A tagged Software is required to have the three associations each with one value.

I continue to have trouble viewing SOFTWARE as data rather than catalog

Then conceptually we should not have a TAGGED_SOFTWARE class since the implication is that it is describing digital bits.

9.26 TAGGED_SPECTRUM

Same questions as TAGGED_SERIES

Why is there no referenced_from Data_Product_Spectrum (why doesn't it exist)?

Need to add it.

9.27 TAGGED_SPREADSHEET

Same questions as for TAGGED_PALETTE except that there are only two possibilities for referenced_from.

9.28 TAGGED_TABLE

Same questions as for TAGGED_PALETTE except that there are six possibilities for referenced_from.

9.29 TAGGED_TEXT

Same questions as for TAGGED_SPICE_KERNEL

9.30 TAGGED_TIME_SERIES

Shouldn't this be subsumed under TAGGED_SERIES?

9. PDS Specification - Section 10 Review 1 - Response 3 Combined

Hi all,

The following contains the compiled responses to the cross checking review of Sections 10 by M. Gordon, A. Raugh, and D. Simpson.

thanks,
Steve

080529 - Review of all three responses - WG

080529 - Response to review comments - Steve
080521 - Review comments - Mitch

080521 - Response to review comments - Steve
080521 - Review comments - Anne

080521 - Response to review comments - Steve
080520 - Review comments - Dick

Review of responses - 080529
---
1) Add "_Explicit" to tagged_file.

2) Standardize Section 9 descriptions.

3) Model data product (Implicit File) and Combined Detached Label (Explicit File) similarly.

4) Move SPICE Kernel to Data classification.

5) Add five classifications to section 8, data classes.

6) Change title of section 7 from "Catalog" to "Context".

7) Drop Section 12.

8) Delete Alias

9) Move *_Map_Projection to context and drop Ancillary

10) Allow the addition of Descriptive Objects to data product, similar to Descriptive Elements.

==========
M. Gordon
==========

At 07:23 AM 5/29/2008, J. Steven Hughes wrote:
Hi Mitch,

Responses below.

Steve

080529 - Response to review comments - Steve
080522 - Review comments - Mitch

At 07:48 AM 5/22/2008, Mitch Gordon wrote:
Sorry these are so late - I've really been struggling to
understand what the specification is trying to tell me.

Mitch

Section 10 comments

In the UML diagram I think the connections from the IDE, DDE, and LSI boxes are inconsistent - see my comments on parallel entities below.

...

The hierarchy diagram is missing Spectra_Cube and Spectrum. Series should be subordinate to Table.

Fixed.

(Repeated comment from my earlier email). I have problems with the 'label' class hierarchies. Since we've broken labels into pieces,
I'm not convinced that the attached, detached, combined detached

distinctions are not implimentation issues. I've put a reduced
hierarchy diagram at the bottom.

Agree. The Attached and Detached distinction has been removed.

Looking at the data products (array, image, qube, spreadsheet,
table) in the hierarchy diagram and their individual table
entries, I get the distinct impression that each is a subclass
of DP_Detached_Label only (not DP_Attached_Label).

Looking for parallel structure - I probably have misconceptions.
If these are incorrect, my assessments are flawed:

* 10.1 Combined_Data_Product & 10.5 Data_Product are
  parallels

They are different classes.

Data Product uses the implicit file object.
Combined uses the Explicit File object.

* 10.2 Combined_Detached_DP, 10.3 DP_Attached_Label, and
  10.4 DP_Detached_Label are parallels

Yes, only label_records is different.

==========

10.1 Combined_Data_Product

Why are there no data_product subclasses? If 10.1 & 10.5
are parallels, they should have the same data product
subclasses.

We could model them it would be messy. For example we could specialize all possible versions of the FILE object, one with Image, one with Table, etc. Essentially it is parallel with data product. We would probably just model the most common examples.

10.2 Combined_Detached_DP (Product with Combined Detached Label)

Why are has_Primary_TDO and has_Secondary_TDO not populated the same as for 10.3 & 10.4?

Can be but am waiting till model settles down.

10.3 DP_Attached_Label (Product with Attached Label)

has_Primary_TDO missing array, spectral_cube, and spreadsheet. Also gazetteer and index tables?

has_Secondary_TDO include History? Alternatively make History a 'Child' DO.

Have combined primary and secondary and added missing TDOs.

10.4 DP_Detached_Label (Product with Detached Label)

same as 10.3

Have combined primary and secondary and added missing TDOs.

10.5 Data_Product

has_Primary_TDO missing Tagged_Array, Tagged_Qube, Spectral_Cube, has_Secondary_TDO add History, or move History from Parent to Child

Have combined primary and secondary and added missing TDOs.

10.6 Data_Product_Array

Why is array a subclass of has detached label?

Need help?

==========

I tried to visualize the hierarchy without label class considerations:

Have combining of detached and attached converges toward your vision.

Combined detached is however still a different class and diverges from your vision.

Specification Sect 10 hierarchy

+ Product
+ + Combined_Detached_DP
+ + + Combined_Data_Product
+ + + Document_Product
+ + + SPICE_Product
+ + + Software_Product
+ + Data_Product
+ + + DP_Attached_Label
+ + + DP_Detached_Label
+ + + + Data_Product_Array
+ + + + Data_Product_Image
+ + + + + Data_Product_IM

My version

+ Product
+ + Data_Product
+ + + Data_Product_Array
+ + + Data_Product_Image
+ + + + Data_Product_Image_Mapped???
+ + + Data_Product_Qube
+ + + Data_Product_Spectral_Cube
+ + + Data_Product_Spreadsheet
+ + + Data_Product_Table
+ + + + Data_Product_Series
+ + + + Data_Product_Spectrum
+ + other_Product

+ + + + Data_Product_Qube + + + Document_Product

+ + + + Data_Product_Series + + + SPICE_Product

+ + + + Data_Product_Spreadsheet + + + Software_Product

+ + + + Data_Product_Table + + + Software_Product

+ + + + + Data_Product_Table_FITS

==========

A. Raugh

==========

At 07:11 PM 5/21/2008, you wrote:
Hi Anne,

Responses below.

Steve

080521 - Response to review comments - Steve
080521 - Review comments - Anne

>Review of Sections 10.7-10.12
>==========
>
>Things I noticed along the way:
>
> In the intro, the second paragraph refers to a different section of the
> document.
>
> Similarly for the second paragraph following the class hierarchy list, and
> the paragraph after the figure. In fact, this paragraph seems to be
> identical to the one preceeding the hierarchy. I don't see any point in
> repeating it.
>

***4 TBD

> Section 10.1 has no "Class Description". Is the concept of a "combined
> data product" actually defined in the standards? How is it different
> from a product?

***1 Combined Detached Label

>
> The list of subclasses in 10.4 seems way too short. Where are the
> DOCUMENTs, HEADERs, COLLECTIONs, etc.?

Document is under Combined Detached Label. There might be two document classes?
Are there Header and Collection products?

>
>
>10.7 Data_Product_Image
>
>The "Class Description" is not specific to this class.
>
>The Data_Product_Image_FITS and Data_Product_Image_VICR subclasses listed
>here don't exist, and shouldn't.

Removed

>
>This seems to imply that images can never have attached labels. That is
>certainly not true.
>

The subclass of attached label image products has not yet been defined.

>Why is "Tagged_Histogram" listed in the Vaule column with no values in the
>other columns?
>

It is a possible secondary TDO.

>
>10.8 Data_Product_Mapped_Image
>
>The "Class Description" is not specific to this class.
>
>I'm confused by the association. Why is the presence of a secondary image
>map object treated differently from the presence of other secondary objects,
>like headers and histograms?

Headers and Histograms are tagged data objects. Image map is not a tagged data object
but more like a catalog object, simply a description.

>
>

>10.9 Data_Product_Qube
>
>The "Class Description" is not specific to this class.
>
>Why is "Tagged_Data_Object" sitting all alone on a line?

***1 Good question.

>
>Why is "Tagged_Header" called out separately as a secondary data object?
>

Removed

>It seems odd to see a QUBE label structure without a HISTORY object. Is it?

Added new association to History. It is no longer a data object.

>
>Following the link to (12.1) Ancillary_Object_Description leads to a table
>which seems to fairly bizarre and completely irrelevant to QUBEs.
>
>

***1 Any subobject is allowed.

>10.10 Data_Product_Series
>
>The "Class Description" is not specific to this class.
>
>Once again, the implication is that all series must have attached labels,
>which isn't actualyl required.

Attached labeled Series have not yet been defined.

>
>The more I see the "has_Secondary_TDO" association listed, the less I like
>it. It seems to pick fairly arbitrary things for "Value" which don't actually
>reflect either the Standards or common use. What's supposed to be going on
>here?

Agree there is a problem and common use needs to be determined.

>
>

>10.11 Data_Product_Spreadsheet
>
>The "Class Description" is not specific to this class.
>
>The assumption of detachment is probably more valid here.
>
>The common problems with associations recur here.
>
>
>10.12 Data_Product_Table
>
>The "Class Description" is not specific to this class.
>
>The assumption of detachment is not valid.
>
>The common problems with associations recur here.
>
>====
>
>You know, if you accept my point that the data object pointer is always
>required even if syntactically ellided, then the distinction between
>attached and detached labels becomes logically insignificant. That might
>save you some trouble in this document.
>

Currently in the model the only difference between attached and detached labels is the use of label_records in implicit_file_attached, ...

The Data object pointer usage is consistent.

==========
D. Simpson
==========

At 10:55 AM 5/21/2008, you wrote:
Hi Dick,

Responses below.

Steve

080521 - Response to review comments - Steve
080520 - Review comments - Dick

At 10:17 AM 5/20/2008, Dick Simpson 650-723-3525 wrote:
Steve:

Review of Sections 10.12 through 10.17.

Dick

- -

Files referenced:
1. PDS3 Spec 0.070916n
2. PDS SR 3.7

10.3 DP_ATTACHED_LABEL

This needs to be spelled out: DATA_PRODUCT_WITH_ATTACHED_LABEL, where "with" is very important. For the past two weeks, I have been assuming this was ATTACHED_LABEL_FOR_DATA_PRODUCT, which is quite different. For dense people like me, the relationships then become very hard to understand. The Class Description actually REINFORCES the wrong interpretation since there are only two sentences and "label" is the subject in each.

Done

10.4 DP_DETACHED_LABEL

See DP_ATTACHED_LABEL above (there are three sentences; "label" or "label file" is the subject in each). Similar comments could be applied to most, if not all, of the Class Descriptions in Section 10.

The only difference I can see between DP_ATTACHED_LABEL and DP_DETACHED_LABEL is has_ILF. In the first it's an Association with value TAGGED_FILE_IMPLICIT_ATTACHED; in the second it's Inherited Association with value TAGGED_FILE_IMPLICIT. This isn't much to work with.

After drilling down, there are two Implicit_File classes, one with label_records.

10.12 DATA_PRODUCT_TABLE

What is the distinction between primary TDO and secondary TDO? The difference is subjective and not required for modeling. It is sufficient to say that a data product can have multiple tagged data objects. PDS SR Chapter 4 is not helpful; it ranks objects according to how one might customarily USE them.

***1 needs discussion since Chapter four seems to provide objective modeling parameters.

Why is the secondary TDO limited to TAGGED_HEADER? I could easily imagine including other objects with a TABLE of binary data. At Stanford, we actually produce histograms, spectra, and time series routinely from our raw data for quick-look diagnostics. It's only because I'm lazy that I never wrapped these with our archival products. Instead we plot them and put the Postscript file into the BROWSE directory with its own label.

***1 we can add the additional objects.

10.13 DATA_PRODUCT_TABLE_FITS

Association requires TAGGED_HEADE_FITS; but I see only TAGGED_HEADER in the document.

***1 I do not understand since I do see TAGGED_HEADER_FITS

Why isn't there a DATA_PRODUCT_IMAGE_FITS?

***1 It could be defined.

10.14 DOCUMENT_PRODUCT

Referenced_from = none seems unimaginative

It is not associated with anything, yet.

10.15 PRODUCT

No comments

10.16 SPICE_PRODUCT

Under Referenced_from, there are five "entities". None exists in the document.

Fixed, removed the "entities".

10.17 SOFTWARE_PRODUCT

Referenced_from = none seems unimaginative

It is not associated with anything, yet.

10. PDS Information Model Specification - L. Huber - Review 1 - Response 2.1

Hi all,

Resend with updates on pointer cardinalities, #5 and #8 below.

The following includes notes from the final review of Lyle Huber's review of the specification document.

thanks,
steve

080626 - Review of responses - WG (AR, MG, DS, SH, RJ, LH, TK, BS)
080623 - Response to review comments - Steve
080623 - Review comments - Lyle

Review of responses - 080626
---

1) Problem resolution overview - Problems found during the review of the specification have been categorized as a) specification error, b) inconsistency, or c) anomaly. An error in the specification is fixed. An inconsistency is defined as a difference between the primary reference source for the specification, the PSDD.FULL, and the PDS standards reference. Inconsistencies are forwarded to E. Rye for resolution. If she determines that a data dictionary change is needed, she will notify the WG so that it can update the specification. An anomaly is a problem or issue in the current data model. These are collected and listed in the anomaly section of the specification document. Anomalies will be further analyzed in a separate task.

During the discussion of whether PSDD is allowed in the CATALOG object, another type of problem was identified. This problem occurs when the validation tools or the EN data engineering staff report errors that the standards documents would not indicate. These types of problems will be researched individually by testing the tool or talking with the DE staff. The finding that a problem exists will at least cause an anomaly to be written and possibly an annotation in the document.

2) Fix QUBE misspelling.

3) ***3 The data dictionary indicates that File_Name is optional in the File Object. This is not consistent with the stds ref. The std refs seems to be correct.

4) The cardinality of "has_index_column" needs to be changed to 1...8.

5) HEADER, TEXT, and SPICE_KERNEL will have required data object pointers.

6) Look into ordering associations and attributes in the class diagram tables.

7) Research note from D. Simpson - There are 31 objects listed in Standards Ref Appendix A. According to PDS_FULL (1r69) PSDD is allowed as optional in all except BIT_ELEMENT, GAZETTEER_TABLE, and INDEX_TABLE. SPECTRAL_QUBE is not in 1r69; if you look in 1r70, you find that PSDD is NOT allowed in SPECTRAL_QUBE. PSDD is also allowed in PARAMETERS. I didn't find PSDD as optional anywhere else in 1r69.

8) Rationale for required pointers. (AR) - My question would be - in what case is it possible for any HEADER or TEXT object to not have a pointer? I don't think there can be one, and here's why:

- Any file on a PDS volume that contains a HEADER or TEXT object must have a label.
- That label must contain the HEADER or TEXT object definition corresponding to the physical object.
- The label must also contain a pointer in some form to indicate the physical location of the data object.

The "in some form" discussion we've had multiple times. Sometimes the form is the explicit pointer statement; sometimes it is derived from the attibutes RECORD_BYTES

and LABEL_RECORDS; and in rare cases like the degenerate labels usually used to label TEXT objects, it is omitted by syntactic convention and taken to be the record immediately following the record which contains the label "END" statement.

At 01:14 PM 6/23/2008, lhuber@nmsu.edu wrote:

Steve,

Here are comments on the Information Model Specification.

- 7.1, last sentence of Class Description should be deleted.

Done. However a wholesale edit of the class descriptions is planned.

- 7.1, does the CATALOG object really have PSDD

Yes. We used the Tool Data Dictionary - version 1R69 - as the gold standard. If an inconsistency is found between the Tool Data Dictionary and the Standards Reference then Elizabeth is notified to resolve the inconsistency and then notify us of any data dictionary changes.

- 7.6, 7.10, 7.17, 7.18, typo on institution_name

Previously identified and fixed.

- 7.9, 7.13, typo on inclusion

Fixed. See comments on propose edit of class descriptions.

- 7.11, 8.4, 8.5, 8.6, 8.7, 8.8, 8.16, 8.17, 8.20, 8.21, 8.26, 8.27,
  8.29, 8.38, 8.41, 8.42, 8.43, 13, what is name_ supposed to indicate?

Good call. Ron found this previously and I thought it had been fixed. The data element "name" is being interpreted as "name_", an internal variable in the tool I am using. TBD fix.

- 7.13, Referenced from Spectral_Cube - is this the only place referenced?

Almost and good call. In the standards reference Image_Map_Projection is an optional object for Spectral_Cube. It is also an optional object for File in the Data Dictionary. These are the only explicit associations currently in the PDS3 standards documents. The issues seem to be, 1) Spectral Cube is not currently defined in the data dictionary. E. Rye will be notified. 2) The File model in the specification document is different than that specified in the PDS3 standards document. In short the direct association from File to Image Map Projection is now through the tagged_file_* class and so will not be in the Referenced from list.

- 7.13, typo on vertical_framelet_offset

Fixed.

- 7.23, No Class Description? Really?

Added one from stds ref. See comments on propose edit of class descriptions.

- 8.18, file_name is really optional in an explicit file object?

***3 Yes, according the data dictionary. Inconsistency to E. Rye.

- 8.19, no PSDD in implicit file object?

***1 Discussion item.

- 8.21, I don't think 20..20 is the proper notation - it should be just 20.

Good call. TBD fix.

- 8.26, Index Column Names are not limited to just those 6 possible values.

The columns of an Index Table are divided into those with required names and those provided by the data provider. These two groups of columns are modeled accordingly. The difficulty is determining the number of required column names since some have two conditional options.

- 8.27, Index Tables are not limited to 7 or 8 columns.

***1 see above. The counts should match.

- 8.34, typo on sample_suffix_name

Fixed.

- 9.20, why is cardinality on has_Pointer none?

***1 Needs discussion.

The value is 1
for 9.21 and 9.22.

Pointers are required for Histograms and History.

Also these have different values in the
Value column.

Different descriptions are required.

- 10.1, typo on Context_Supplemental

Fixed.

- 13, typo on easternmost_longitude

Fixed.

- 13, typo on institution_name

Fixed

- 13, typo on sample_suffix_name

Fixed

- 13, typo on vertical_framelet_offset

Fixed

- 13, typo on westernmost_longitude

Fixed

Lyle Huber
PDS Atmospheres Node

11. PDS3 Specification - T. King Review 1 - Response 4 - Final

Hi all,

The following includes the final responses to Todd King's review of the specification document.

thanks,
steve

080717 - Review of responses - WG (AR, MG, DS, SH, RJ, CI, BS)

080703 - Response to review comments - Steve
080702 - Response to review comments - Steve
080630 - Response to review comments - Steve
080626 - Review comments - Todd

Review of responses - 080717
---
1) The WG decided that the items in each Section would be sorted alphabetically. The class hierarchy listings are included at the beginning of each section and in the class definitions.

2) The WG decided to reorder the sections. 1 - Data Object and the simplest product component classes, 2 - Data Descriptions, 3 - Tagged_Data_Objects, 4 - Products, 5 - Context, 6 - Operational.

3) Add text to section 5 - Terminology. - "An association has one direction. The association in the opposite direction is called the inversion relation and is sometimes denoted with a postfix "_I" as in "has_Instrument_I".

4) Determine the meaning of "><" in the UML class diagrams. Omit if possible.

5) In the Class Definition tables change the title of the "Value" column to "Value/Class".

6) Add "units" to the specification data dictionary.

-- Response 1 --

At 12:28 PM 6/26/2008, Todd King wrote:
Hi all -

My apologies for sending this out so close to the telecon. I just didn't
have time to type up my comments until just a short time ago. Deadlines are
wonderful motivators.

PPI (Todd) comments on PDS Information Model - June 9, 2008 draft.

4. History
Statement of "de facto standards representing common contemporary..."
is very odd. I didn't get a lot of it.

INTRO

5. Terminology

Sentence "Please consult the Glossary for definitions whenever there is confusion."
How will anyone know they are confused? The should be "Please consult the Glossary for current definitions".

INTRO

Define "Abstract Class". It is used in the text, but not defined.

Good point. TBD

In "Cardinality" paragraph. Change "allowed to an" to "allowed for an".

INTRO

"Entity" paragraph. drop word "specific". Entity is composed of attributes or associations.

INTRO

6. Document Contents
In the paragraph describing the tables use field titles rather than relative position.
For example, change "the third column" to "the 'Card' column". If the table changes the text
will still be valid.

INTRO

The use of "Ind" column and "O" and "R" is not common practice. While some hold that a cardinality
of "0..1" or "0..*" is not "valid", if you have to add a column to indicate the same thing while limiting
minimum cardinality to 1, then there must be valid instances for a minimum cardinality of 0.

Agreed that it is not common practice. The problem is that the PDS concept of "optional" attributes should not overload the object-oriented meaning of cardinality. A cardinality of zero formally means that no value is required. To add to this meaning something like "no attribute is required" makes the concept ambiguous.

The concept of optional attributes is generally discouraged in formal data modeling, especially at the implementation level, but if allowed it should be indicated using a separate notation.

In the PDS archive, catalog file and product labels do not have optional attributes. During the product design phase, "generic objects" are used to create "specific objects" and these are documented in the SISes. The label files generated from these "specific objects" must have the same set of attributes and each attribute must have a value. Values of N/A, UNK, etc are allowed. Therefore, a valid model of a PDS product can not have an attribute cardinality with its min value being "0".

The paragraph that starts with "The PDS has defined..." should be under the heading of "PDS Idiosyncrasies"

INTRO

Same paragraph. Shouldn't "object" be "Object" since they are formal entities?

INTRO

Same paragraph. The statement "Where a specific object has required attributes and sub-objects, a
generic object has optional attributes and sub-objects.". Don't kinds of objects has optional and required?

Not by definition. Any inconsistencies that were found for added to the anomaly list or fixed.

"The Token PSDD is used to indicate..." This means the data model is not
robust. While this has been
an aspect of PDS3, the described data model is PDS3.
Why carry invalid
modeling practices forward?

This is related to the discussion on "optional" attributes above. Another way to understand the issue is to consider two possible PDS3 models.

1) The first is the "robust" PDS3 model as implemented. This model would have a class for each product type in the archive. There would be thousands of classes but no need for "optional" attributes or PSDD.

2) The second is an "augmented" model of PDS3 as practiced. This requires the understanding that PDS data modeling has two phases. The first phase uses modeling guides (generic objects) to define specific objects. Phase 2 then follows with the documentation of the specific objects in SISes and the generation of products.

We can generate the first model. It would certainly be much simpler to accomplish than the second but I am not sure how useful it would be.

Actually the new label design tool outputs a label_template. These are "specific objects" in the ODL modeling sense or product classes expressed in ODL in the specification sense.

7. Context Description Class.

Paragraph that starts with "The catalog model is illustrated...". Aren't
inverse associations inferred
from the semantics of the forward (direct) associations?

No, each direction is explicitly defined.

Why are
associations double ended?

This is an artifact of the UML generation tool. It does not relate to a semantic but I will look into removing it.

General comments for Classes sections:
The sections should be organized from the broadest class to the most specific class. So, for this section
the order should reflect the tree. Otherwise referenced items are sometimes not defined before they are
used.

Yes, we have discussed the ordering of the items within a section and the sections themselves. Once the specification nears its completed form, we will discuss this again.

7.1 Catalog
The reference to "VOLUME" in the class description is not relevant.

I do not understand. CATALOG is associated to VOLUME in the model. So the link is valid. CATALOG is a required object of VOLUME so the model is consistent with the standards.

The "Association" names don't map to actual "classes". For example, "has_Catalog_Data_Set" points to what?

"has_Catalog_Data_Set" is an association between the CATALOG class and the DATA_SET class. This association is explicit in the PDS3 standards though the "has optional" or "has required" objects ODL construct.

There is no "Catalog_Data_Set" class.

The PDS has few association names, for example "has optional" and "has required". Using these as the only association names in the model was problematic. We have the option of now providing more meaningful names for associations. For the specification the formation rule for association names was to use the "to" class in the association name, as in "has_Data_Set". However since each name must be unique, and has_Data_Set had been previously used and had different characteristics, the "from" class name was concatenated. We can certainly review and edit the associations names as the WG wishes.

"Data Set" is not a subclass of

"Catalog".

Agreed, it is not.

The "value" gives a hint that
it points to a class, but the specification is very incomplete.

I do not understand. Should the hint be made more explicit? Does incomplete mean that not all links work?

7.2 Context_Child
The use of "abstract class" is really a "meta-class".

An abstract class is a class that can not be instantiated. The data modeler has the choice of designating a class as abstract.

Most of our abstract classes are parent classes that do not have attributes. (However this is not precluded.) They are being used to group our classes based on function or some other characteristic.

These characteristics could be modeled as a class_type and included in the model as an attribute of the class. But in the object-oriented paradigm the existence of a *_type attribute suggests that subclasses should be defined.

These abstract_classes could also be considered a classification scheme. But classification schemes are simply predefined sets of classes.

The class has not attributes or behaviors. Why is it a class? Its degenerate
and should be pruned.

See above.

Likewise with Context_Core, Context_Description, Context_Description, Context_Supplemental.

See above.

7.6 Data_Set
This is not the PDS3 Data Set Object. Some of the attributes are from the DATA_SET_INFORMATION object,
some from a HOST object, some from MISSION object, Some from TARGET object. Yes it also as associations
to the each objects (except DATA_SET_INFORMATION).

The rationale for collapsing the ODL implementation model was reiterated during the review of Lyle's comments.

The description also runs
off the page. How do you plan
to create a printed version?

I do not understand. I assume that you are looking at the PDF version.

The text description wraps in my PDF versions. I would like to see an example where it runs off the page.

However the Table definitions do sometimes run off the page. This is due to my limited LaTEX experience. I welcome hints but have not had the time to find the solution.

7.8 Data_Set_Collection
Attributes are from both DATA_SET_COLLECTION_INFO and REFERENCE, but it has
a has_Reference.

I do not understand.

7.11 Directory
The sentence "It is a required sub-object of the VOLUME object for volumes delivered on sequential media".
The original text referred to "Tape". This rewrite implies a broader use. If
we are to re-write a definition
why not eliminate tape all together.

Good point. Quick "cut and paste" operations were done from the std ref for the specifications class descriptions. As mentioned during Lyle's review a wholesale cleanup of class descriptions is planned.

The Association of "has_File" does not reference an defined class. There is no "File" class.

***1 Good call. File was renamed to File_Explicit.

has_File should be optional.

It is "required" in the standard reference and PSDD.

7.13 Image_Map_Projection
This is product level information and product specific. Why is it in the "context" section?

***1 Currently there is no "level" differentiation in context. This is a good discussion point.

While its "context" for an observation its not comparable to other items in section.

The description is edited form the original which has changed the meaning. The sentence
that beings with "The map projection information.." tries to justify why it exists by arguing it
reduces "data redundancy". Why is that part of a definition?

There is a wholesale cleanup of class descriptions planned.

This is as far as I got with the time I had. When flipping through the document I noticed many issues

similar to ones described.

Any systematic problem identified above will be fixed. Wholesale cleanup of the descriptions is planned. Any remaining specific problems will hopefully be found by the additional reviews.

For example under "7.14 Instrument" there is an
association called "has_Instrument_I"
which is to have a value of "Data_Set". This is inconsistent.

I do not understand how this is inconsistent. The association name "has_Instrument_I" is short hand for the inverse of "has_Instrument". Both relationships are implemented in the Data_Set_Host ODL subobject.

This document
needs a great deal of additional work
and review.

I agree in terms of class descriptions, association naming, and a few other similar areas.

However since most recent reviews are finding relatively few significant modeling errors I think that the model is pretty stable as far as its basic components.

Granted the definition of product and its subclasses are somewhat controversial. This is to be expected since product was never formally defined in the PDS3 standards.

thanks,
steve

-Todd-

-- Response 2 --

> >I'm confused here. The "inverse" and "double-ended" comments are the
> >same issue, but the responses are in conflict.
>
> Sorry, I was not clear. Inverse relations and the "double-ended"

> artifact are separate issues. In the UML, a relationship and its
> inverse can be depicted as either one line with two pairs of
> cardinalities or two lines, each with one pair of cardinalities.
> However, the meaning is the same. The difference is whether each has
> a unique name or label, assuming all other meta-attributes are equal.
>
> The "double-end" artifact does not change in either case. I am
> looking into what it means, if anything.
>

All true, but "lines" and the cardinality are currently represented in the
class as attributes. Isn't this an implementation issue? That is, how to
represent an association. It may be different depending on the

Please explain why this must be an implementation issue. I see the issue of how to represent an association as a modeling issue. If we accept that associations are components of models then they must exist in conceptual, logical, and physical (implemented) models. How they are represented depends on the flavor of modeling notation one chooses.

Also please explain your view that "lines" (relationships) are represented in the class as attributes. (Note that in certain modeling paradigms relationships are modeled as attributes with an "instance typed" value.)

I see Relationships in the UML diagrams being represented by lines.
I see Relationships in the specification tables being titled as "Associations", a specific type of relationship.

***1 Review introductory text regarding this issue. Specific suggestions are welcome.

implementation.

> >The comment here is based on best
> >practices. Singly-linked lists are better then doubly linked list
> >because of the update and referentially integrity issues.
>
>
> These are implementation issues. However now I am confused. The
> original issue was uni- and bi-directional relations. The concept of
> linked lists is implementation. Update and referential integrity are
> implementation issues.
>
>

>
> > > >The "value" gives a hint that
> > > >it points to a class, but the specification is very incomplete.
> > >
> > > I do not understand. Should the hint be made more explicit? Does
> > > incomplete mean that not all links work?
> > >
> >
> >Aahh! This is a great example of good response to a review
> comment. Let
> >me expand. The text in chapter 6 the describes how to read
> the classes
> >table states the following: "Significant or unusual constraints or
> >attributes values are listed in the forth column." The
> "forth" column
> >is the "Value" column.
> >
> >Using "7.15 Instrument_Host" as an example. In the "Attribute" the
> >values listed for "instrument_host_type" are what?
>
> I might be confused on the issue but SPACECRAFT is one value
> that is listed.
>

By "what" I meant a literal string, a class name, or named reference?
Which the following text tried to illustrate. Values under "Association"
appear to be of a different type then values under "Attribute".

Thanks for the clarification.

***1 This information is in the model database. It can be made more clear in the specification. Specific suggestions are welcome. Note that the information will be in PDS3 nomenclature, for better or worse.

> > I can infer, from my
> >understanding of PDS, that the "instrument_host_type" is an
> enumeration
> >and the values list are possible values.
>
> Yes
>
> > By this is not clear in the
> >specification. Also I would expect some indication for

> >"instrument_host_id" that it be a valid identifier.
>
> Good point. This would probably required a formation rule. We have
> not addressed formation rules for values in the specification, yet.
>
>
> > The specification seems to be relaying on
> >class names to indicate data type. A formal specification should be
> >explicit.
>
> Please explain your issue. The specification defines a class
> explicitly in terms of its attributes and associations. It also has a
> name but the name is only a label which hopefully is human friendly.
> If not we can rename it.
>

I was using "class name" in a PDS context. For example named with the
pattern "*_id" are identifiers which have a restricted character set and
must be well-formed. "*_name" also has a formation rule as does "*_time".
Without type information all that can be assumed is that all attributes are
of a single generic type.

Thanks for the explanation. The content of the PSDD is present in the model database.
The specification data dictionary is generated using the pertinent information from the
PSDD.

***1 Discuss the inclusion of additional information from the PSDD into the
Specification data dictionary.

>
> >In the "Association" section of the table, the value gives the class
> >name, but is that a value or an implementation name for the
> class? Its
> >ambiguous.
>
> Please explain your issue. An association is a type of relationship
> between two classes. The value mentioned is the name of the class
> explicitly defined in the specification. If it is not defined then it
> should be and would be an error in the specification. I do not
> understand what "implementation name" means in this context.
>

The issue is that this is not stated anywhere in the specification.

***1 Discuss an appropriate update to the introduction.

>
> > >
> > > > For example under "7.14 Instrument" there is an association
> > > >called "has_Instrument_I" which is to have a value of
> "Data_Set".
> > > >This is inconsistent.
> > >
> > > I do not understand how this is inconsistent. The
> association name
> > > "has_Instrument_I" is short hand for the inverse of
> > > "has_Instrument". Both relationships are implemented in the
> > > Data_Set_Host ODL subobject.
> >
> >This short-hand is not explained in the document. The
> description needs
> >to explain what an "inverse" "has_*" association is. Its not
> a common
> >modeling concept.
>
> Please explain your issue further. Inverse relations are common
> modeling concepts. Relationships have names but they are simply
> labels. They should be human friendly. A general cleanup of names and
> descriptions is planned. But they are just labels.

This is similar to the value associated with an "Association" (the class name). You know that a "has_" association that ends in _I is an inverse association, but this is not explained in the document. As a new reader of the spec. I may not guess correctly that _I means inverse.

***1 Discuss the cleanup of relationship names.

-- Response 3 --

The following are queued for discussion at the next telecon.

***1 PSDD

***1 handling of units for values

***1 data types of values.

***1 introduction of phantom (or category) classes which are used to group classes.

12. PDS Specification - D. Simpson Review 6 - Response 2 - QUBE Cross Check

Hi all,

The following includes the final responses to D. Simpsons's review of the specification document.

thanks,
steve

080717 - Review of responses - WG (AR, MG, DS, SH, RJ, CI, BS)
080716 - Response to review comments - Steve
080716 - Response to review comments - Elizabeth
080716 - Review comments - Dick

Review of responses - 080717
---
1) Add anomaly. - "QUBE - Special requirements are imposed by the ISIS system but the Stds Ref leaves open the question of whether these are also PDS requirements. The text for PDS users should be clarified.""

2) ***3 - The Std Ref should clarify that the maximum of 6 dimensions for the PDS QUBE are ISIS and not PDS limitations.

3) Re-model QUBE suffixes to address the implicit association that exists between 2 or more named suffixes. (I.e. One of each must exist)

4) Add File_State to QUBE.

5) Add anomaly. - "The current model for products is generic. It does not have subclasses for Image, Table, etc. Therefore it is difficult to determine the required tagged_data_objects and their associated data objects for a specific product type."

At 10:58 AM 7/16/2008, Elizabeth D Rye wrote:

Sorry, forgot to CC the group on my response to Dick.

Elizabeth

Begin forwarded message:

From: Elizabeth D Rye < Elizabeth.D.Rye@jpl.nasa.gov>
Date: July 16, 2008 8:54:37 AM PDT
To: Dick Simpson 650-723-3525 < rsimpson@magellan.stanford.edu>
Subject: Re: QUBE Cross Check

Hi Folks,

Sorry to jump in on this, but the QUBE in the subject caught my eye.

A. Optional elements: psdd.full lists PSDD, but it does not appear in
either PDS3 Spec or PDSSR.

Needs to be fixed.

Fixed in Spec.

***3 To Elizabeth

B. PDSSR says "special requirements are imposed by the ISIS system" but
leaves open the question of whether these are also PDS requirements.
If not, I would recommend deleting this and other ISIS-specific
information from PDSSR. In any case, the text for PDS users should be
clarified (Elizabeth?). The PDS Operator is listed as a source for
ISIS documentation, which also seems irregular.

I disagree. We went to great lengths in developing the SPECTRAL_QUBE standard to make sure that it would be possible for a data provider to provide a science product that was compliant with both PDS and ISIS standards. Since the ISIS folks have zero interest in explaining how to do this, it is up to the PDS to describe the details. Patty made it very clear in this chapter which requirements came from the PDS and which came from ISIS. If there are any ambiguities remaining in the chapter, we need to fix them, not remove the references to ISIS completely.

***1 Needs discussion.

C. Naming conflicts between QUBE elements and PDS nomenclature standards are a recognized anomaly (PDSSR page A-78).

I will hold to my dying day that the QUBE object is not a PDS3 object, but rather a holdover from PDS2.

D. Cardinality of AXIS_NAME should be 1..*. QUBE is multi-dimensional; the number of dimensions can be as small as 1 and no upper limit is defined in either PDSSR or psdd.full. However, PDSSR appears to limit AXIS_NAME and CORE_ITEMS to 6 values.

Yes. That is because the QUBE was both a PDS and an ISIS construct, and ISIS placed an upper limit of 6 on the cube. Hence my old argument that the PDS2/3 QUBE needs to be replaced by a more generalized PDS QUBE of unlimited dimensions. The PDS3 SPECTRAL_QUBE was meant to completely replace the old PDS2/3 QUBE. I personally advocated for deprecating the QUBE; Dick, you were the main opponent of this approach.

***1 Axis Name cardinality set to [1,6]

E. Cardinality of CORE_ITEMS should be equal to the cardinality of AXIS_NAME.

Cardinality set to [1,6]

F. PDSSR indicates cardinality of SUFFIX_ITEMS should be 1..6; but a QUBE with more than three dimensions cannot have suffix areas. So we should change the cardinality to 1..3, add "O" to the Ind column, and flag the 4..6 question as an anomaly for Elizabeth.

SUFFIX_ITEMS cardinality set to [1,3].

***1 SUFFIX_ITEMS is "required" so in the case of 4..6 dimensions it seems that its value must be 0.

***1 SUFFIX_ITEMS for more than three dimensions is an anomaly?

Be careful. I agree there's an inconsistency there, but you can't make changes to the QUBE without the approval of the ISIS folks. This is the entire reason we developed the SPECTRAL_QUBE. Remember that this object is jointly owned and controlled by the PDS and ISIS, and after years of arguing with them, it became eminently clear that the ISIS team is NOT interested in changing it.

G. Optional QUBE suffix areas are shown in the QUBE definition as relationships has_Qube_Band_suffix, has_Qube_Line_Suffix, and has_Qube_Sample_Suffix. This can work for the Standard ISIS QUBE if an implicit relationship is assumed between the specific suffixes and the QUBE axes. More generally, such as when there are fewer than three axes or when the QUBE axes are not named band, line, and sample, this breaks down. In simple terms, which suffix goes with which axis, and how do we know? The fact that the cardinality of each association is 1..* adds to the confusion; I would expect each to be optional with cardinality "1".

***1 It seems that we have two QUBEs.

1) The "implicit" relationship can be made "explicit" several ways in the model, one is by creating a containment class. So for the [Band, Line, Sample] case we create a Qube class with a single "has_suffix" relationship to a containment class which contains the necessary suffix subclasses.

2) The non-named case would have a generic suffix class.

These alternatives can be modeled as either a base QUBE class and two subclasses or as two separate QUBE classes. Either way we are probably talking about at least two models for the QUBE to address this issue correctly.

H. Anomaly: If a suffix pixel occupies fewer than the allocated 4 bytes, there doesn't seem to be any way to specify which bytes are occupied (right-justified, left-justified, centered, ...?).

***2 Anomaly: If a suffix pixel occupies fewer than the allocated 4 bytes, there doesn't seem to be any way to specify which bytes are occupied (right-justified, left-justified, centered, ...?).

I. According the PDSSR (page A-80), there can be multiple values for *_SUFFIX_NAME, *_SUFFIX_UNIT, etc. This is presumably because the suffix is allowed to be "heterogeneous" (page A-77). (1) The cardinality for attributes in PDS3 Spec 8.32, 8.33, and 8.34 is always "1," which is incompatible with heterogeneity.

***1 Discussion

(2) If *_SUFFIX_NAME applies to the entire suffix, why is there more than one name?

Because it's not the name of the suffix, it's the name of the quantity stored in the suffix. The "*" is the name of the suffix.

(3) How are the other keywords matched to their axes? This gets messy if there are three suffixes, each of which has three dimensions

Again, this is why we developed the SPECTRAL_QUBE, which uses groups to clean up these relationships.

***1 These "implicit" relationship can be made explicit by creating a containment class....

J. PDSSR says ISIS requires that a QUBE be associated with a HISTORY object; so has_History_object must at least be optional in 8.31.

***1 Since HISTORY has been modeled as a class and as a component of Tagged_History_Object then the relationship would in the QUBE product class.

Several other objects are listed as optional on page A-78.

Ditto for these objects.

This gets messy because the HISTORY object specified is not clearly a PDS object. You'll note that there are no required or optional elements for the PDS HISTORY object. This absolutely needs to be fixed under PDS4.

K. FILE_STATE is required in an implicit FILE object under ISIS. (1) FILE_STATE should be at least an optional attribute in 8.31.

***2 Anomaly? FILE_STATE seems to be an attribute of QUBE and so would be included in 8.31.

(2) There
appears to be no way to get from FILE_IMPLICIT to QUBE.

***1 Try 10.1 Data_Product -> 9.21 Tagged_File_Implicit -> 9.13 TDO_Core -> 9.31 Tagged_Qube -> 8.31 Qube.

Since the QUBE Product has not been modeled then there is no path through explicit associations. You have to take an alternate path through the subclass relationship at 9.13 TDO_Core.

L. Some information (e.g., BAND_BIN* and IMAGE_MAP_PROJECTION) is organized into GROUPS in ISIS, so that should be an option for PDS.

There are other uses for groups besides this. The GROUP has been re-instated as a legal construct in PDS3, so yes, it should be defined in the specification.

***1 Replace "group" with "class". TCMIII.

Sorry for butting my nose in. Hope this helps.

Your input was additional useful information.

thanks,
steve

Elizabeth
--
Elizabeth D. Rye
Phone: (818) 354-6135
Email: Elizabeth.D.Rye@jpl.nasa.gov

13. PDS Specification - D. Simpson Review 7 - Response 2 -
Spectral QUBE Cross Check - Final

Hi all,

The following includes the final responses to D. Simpson's review of the Spectral QUBE.

thanks,
steve

080724 - Review of responses - WG (AR, MG, DS, SH, RJ, CI, BS)
080722 - Response to review comments - Steve
080722 - Review comments - Dick

Review of responses - 080724
---
1) ***3 SUFFIX_BYTES will be set to "optional" to be consistent with PSDD. However the PDSSR has SUFFIX_BYTES as "Required".

2) ***3 Table in standards reference (A25.4.5) seems to be wrong since it allows 1, 2, and 4 as values. However the narrative mentions that an ISYS constrain requires that it be 4.

3) ***2 Spectral_Qube - Conditional keyword anomaly - SUFFIX_BYTES is a required attribute if suffix planes are included.

At 10:41 AM 7/22/2008, Dick Simpson 650-723-3525 wrote:
Steve and Others:

The SPECTRAL_QUBE seems to be in better shape (vis a vis the PDS3 Spec) than the QUBE. But there are still some confusing parts. These may reflect the fact that we are pushing the model envelope with groups, conditional attributes, and suffixes with similar names. This is what I have. Note that #2 has multiple parts, some of which may be answers to earlier questions.

Dick

- -

Comments on SPECTRAL_QUBE (8.39):

Sources:
PDS Standards Reference (PDSSR) version 3.7
psdd.full version 1r70
PDS3 Specification version q

1. BAND_BIN is listed as a required object in pds.full (with object_type = generic_group), as a required group in PDSSR, and as a required attribute (with association has_Band_Bin) in the PDS3 Spec. Why different, and why attribute? I think because BAND_BIN is simply shorthand for a set of attributes, yes?

***1 Good call. BAND_BIN and the *_SUFFIX items were modeled as attributes. This was a typo. They are now modeled as classes. Since "group" is an ODL construct (i.e. implementation), it is replaced by "class" in the conceptual model.

2. Band_Suffix, Line_Suffix, and Sample_Suffix are listed as optional objects in pds.full, as optional groups in PDSSR, and as optional attributes (with associations has_*) in PDS3 Spec. Why all different, and why attributes?

See above.

2a. In PDS3 Spec I think has_Band_Suffix, has_Line_Suffix, and has_Sample_Suffix provide the same shorthand as has_Band_Bin above; but this only gets the attributes into the SPECTRAL_QUBE specification.

See above.

2b. The suffixes themselves need to be added under ASSOCIATION: has_ Spectral_Qube_Band_Suffix, has_ Spectral_Qube_Line_Suffix, and has_Spectral_Qube_Sample_Suffix would be my preference. Section 8.40 puts us on the correct track via a phantom object. Unfortunately the actual subclasses listed are Band_Suffix, Sample_Suffix, and Line_Suffix -- which are the names already taken by our shorthand.

This all needs to be reworked.

2c. But subobjects usually bring keywords with them; so why are the subobject keywords listed separately under SPECTRAL_QUBE ... ? Subsuming? In any case, the modeling notation is under challenge here.

2d. Band_Bin is listed as a subclass of Spectral_Qube_Bin_Suffix. But Band_Bin relates to the core, not to a suffix. It should not be included here.

The "phantom" class has been removed. There seemed to be no plusses and at least one minus for its existence. The *_Suffix class names remain as specified in the PSDD and Stds Ref. The association names remain as versions of has_<class name>.

3. SUFFIX_BYTES is a required attribute if suffix planes are included (PDSSR); but it is listed as optional in pds.full and as required in PDS3 Spec. Conditional keyword anomaly.

***3 SUFFIX_BYTES is a required attribute if suffix planes are included PDSSR but it is listed as optional in pds.full

***2 Spectral_Qube - Conditional keyword anomaly - SUFFIX_BYTES is a required attribute if suffix planes are included

4. Many attributes can be multi-valued; but all are shown in PDS3 Spec as having cardinality 1. When multi-valued, they must be sequences (not sets); is this the difference? If so, we need to convey the distinction to readers in Chapter 6.

***1 If multi-valued in the Std Ref then they should be in the Spec. Many have been set to 3..3 as per Std Ref. Probably needs checking.

5. AXIS_NAME is a required attribute, and three values are required (a sequence). It is shown as having cardinality 3. How is this different from the situation in 4 above? Ditto CORE_ITEMS, SUFFIX_ITEMS, ...

It is now consistent.

6. ISIS_STRUCTURE_VERSION is an optional attribute in PDSSR and psdd.full; but it is missing from the list in PDS3 Spec.

Added.

14. PDS3 Specification - C. Isbell Review 1 - Response 2 - Final

Hi all,

The following includes the final responses to C. Isbell's review of the specification document.

thanks,
steve

080731 - Review of responses - WG (AR, MG, DS, SH, RJ, CI)
080730 - Response to review comments - Steve
080730 - Review comments - Chris

Review of responses - 080731
---
1) The WG decided on a "best practice" for modeling the cardinality of a class attribute in the specification model. a) If the PDS3 standards do not specify the cardinality's upper bound then the cardinality will be presented as "1..*". b) If the standards explicitly specify that the cardinality has no upper bound then the cardinality will be presented as "1..*" and an indicator "K" (i.e. Known - upper bound specified as "*") will be included the "IND" column. c) If a specific upper bound (less than *) is specified, then it will be presented as such. c) If the standard value set for an attribute has ODL sequences as values then the cardinality will be "1". (e.g. band_sequence).

2) ***2 The use of sequences as standard values is an anomaly. For example the PSDD defines Band_Sequence with general_data_type = CHARACTER and has a standard value list consisting of ODL sequences, each containing a unique permutation of three values.

At 03:07 PM 7/30/2008, Christopher E Isbell wrote:

Steve and group,

Here is an initial review of the specification. I have reviewed the IMAGE specification which spawned a few MISC items further below. I intend to continue with MAP_PROJECTION, QUBE, and SPECTRAL_QUBE section review.

Chris

Version 0.07091r

8.24 Image

Attribute

band _sequence

I know previous reviews have addressed this but for the record ... In PDS context, how are we defining cardinality? If we follow a mathematic definition ("the number of elements in a set or group"), then it seems cardinality is "3" for this attribute. If we consider a (sequence) a single value, then cardinality is "1". Anomaly? Hopefully additional discussion will clear this up for me (us?).

***1 Needs discussion. There was a previous discussion on this topic and Dick understood that if multiple values were provided in a sequence then the cardinality was 1. The single sequence is considered the value.

See above for WG resolution.

I propose that we consider the use of a sequence (or a set) for multiple value assignment as an implementation issue. This means that in the conceptual model the cardinality is simply the number of values to be allowed, regardless of whether a set or sequence is used. We leave it to the implementor of the model and the choice of implementation language as to how the assignment of multiple values is to be handled.

description (and other 'free txt' fields)

Is there any need to recognize/identify "free text" values within the specification?

***1 Needs discussion - Assuming that this refers to the Description (and Note) pointer.

WG consensus was that "free text" fields do not need to be identified.

Misc

Are we supposed to note items like the following?

Not represented in current version of specification

filter_name
producer_institute_name
spacecraft_name

CENTER_FILTER_WAVELENGTH
BANDWIDTH
PRODUCER_INSTITUTION_NAME
band_name
null

Within some existing products, not in current DD nor specification

source_image_id
LOW_REPR_SATURATION
LOW_INSTR_SATURATION
HIGH_INSTR_SATURATION
HIGH_REPR_SATURATION

Since we are using the PSDD as the source of the PDS3 model and since these are not explicitly listed in any object definition, they are not included in the specification model. When we consider the CORE elements of the model for PDS4 and depending on the approach taken (e.g. best examples) then some might be included.

++++++++++++++++++++
Chris Isbell
Cartographer
U.S. Geological Survey
2255 N. Gemini Drive
Flagstaff, AZ 86001
cisbell@usgs.gov
Phone: 928-556-7211
++++++++++++++++++

15. PDS3 Specification - T. King Review 2 - Response 3 - Final

Hi all,

The following contains the WG response to T. King's continued review of the specification document.

The assumption made is that the review was of Version 0.070916p

thanks,
steve

080731 - Review of responses to end - WG (AR, MG, DS, SH, RJ, CI)
080724 - Review of responses up to and including 11.4 - WG (AR, MG, DS, SH, RJ, CI, BS)
080718 - Response to review comments - Steve
080718 - Review comments - Todd

Review of responses - 080724

---

1) Bit_Element to be removed from specification since it is not defined in the PDS3 standards. Previous Bit_Element anomaly remains. Done.

2) Add an indicator in the Value column for attributes with valid value lists (Enumerations). In the data dictionary, list up to 25 valid values. Suggestion was made to generate a specification with all valid values when V3.8 of the stds ref is released. This provides a complete specification. Done

3) Update the TBD descriptions for the abstract classes recently added. Done

4) A suggestion was made to make Gazetteer_Column and Index_Table_Column subclasses of Column. However on inspection this would complicate the model since there is not a base column class. Descriptions to be updated in any case to explain why they exist separate from Column.

5) ***2 (Anomaly) Figurative constants (e.g. unknown_constant) should be modeled in Column and not Index_Table.

6) Ask Todd to expand comments on 9.8 TDO_CORE [polymorphic, generic]. Done

7) WG decided that all tagged data objects should have one required pointer and one required data_object as the normative model.

***2 (Anomaly) Tagged_Text_Object has no required pointer.

***2 (Anomaly) Document allows more than one data_object.

***2 (Anomaly) Implicit_File does not have a pointer.

8) Delete "END" as a class attribute. Done

9) "part_of_data_set" -> "relates_to_data_set" Done.

10) Any "general" association that uses "has_" should be renamed to use "ref_" (i.e. references) to distinguish general from {aggregate, composite} associations. Done.

-- 080731 --

11) The WG consensus is that the "unification" section's narrative and UML diagrams should be updated. The question as to whether it should remain as part of the document was tabled. Add OAIS RM to history section as a source document.

12) Update data dictionary section a) change name to Specification Dictionary, b) Modify narrative to explain that the data dictionary is a subset of the PSDD and contains only those data element used as attributes in the class definitions. c) it also include association and valid value definitions, and d) define PSDD.

13) ***2 - Add an anomaly that for defining data elements the current data type specifications are not sufficient.

14) Request Todd to clarify and provide examples for "14 Cardinality - The definition is correct, usage of cardinality in class definitions do not adhere to this definition."

15) Remaining data dictionary issues - a) In general the PSDD information has been captured in the specification as is. This is consistent with the purpose of the specification model which is to capture the PDS3 model as it is currently documented. Issues associated with the current data dictionary content, especially where the content is clear, are not in general applicable for this review. These issues are captured in this document for future reference. b) Since the specification dictionary is a subset of the PSDD and its purpose is to support the information model, there is no need to present all data element definition information such as min/max values, standard value types, or formation rules. The PSDD can be referenced for such information. When and if the specification is used for driving an implementation this information, which in the info model database, can be extracted and exported to the implementation notation. c) WG consensus is that context_dependent is well defined and used consistently in PDS3. d) The name of the data dictionary is being changed to specification dictionary. It will contain attribute, association and some valid value definitions.

At 12:34 PM 7/18/2008, Todd King wrote:
Hi Steve and others -

My previous review left at the end of Chapter 7. This is review is for the remaining chapters.

There is a persistent "name_" as a attribute in many classes. This should be "name". I think this may already have been mentioned by Lyle.

Fixed.

8.2 Band_Bin and 8.3 Band_Prefix
Referenced From is "none". Aren't these used somewhere. Qubes?

Good call. 8.2 Band_Bin was modeled as an attribute and not a class.

Assume you mean 8.3 Band_Suffix. Same error.

Fixed.

8.4 Bit_Column
The description is completely different from the one in the PSDD. The one in the PSDD is more accurate.
Not sure why it got changed if we're capturing at this point.

The class descriptions were gleaned from the Std Ref. The WG needs to decide when and if a wholesale cleanup of the class descriptions is to be done.

Code:>>WCDC

8.5 Bit_Element
Has no attributes in the PSDD. This is indicated as a anomaly, but the spec defines a class for Bit_Element with attributes. If it doesn't have a complete definition in PSDD does it need to exist? If it defined it appears that Bit_Column is a sub-class of Bit_Element.

Previously captured as anomaly 010_080204_002_Bit_Element_NotDefined.

8.7 Column
The attribute "data_type" is an enumeration, but no values are indicated in the "Value" column.

***1 Discussion needed on model presentation.

The values for data_type and all attributes with "enumerated" valid values are provided in the specification data dictionary together with their PDS3 general_data_type.

The "Value" column when used for an attribute indicates the values of "_type" keywords that suggest a class hierarchy (e.g. instrument_host). This needs to be made clear in the introduction and made consistent in the document if this approach is to be used.

The use of the data dictionary for enumerate valid values reduces the duplication of valid value lists.

The conceptual model will be used to create "enumerated" lists in the derived logical and physical models during implementation, if the target formalism requires them.

Code:>>ENUM

So, "data_type" is inconsistently defined when compared to other class definitions in the spec.
Such as 7.15 Instrument_Host.
Same is true for "data_type" in 8.17 Field, 8.20 Gazetteer_Column, 8.23 Histogram.

See above.

** General comment **
Some typing information and reference to enumerations is critical for evaluating the class definitions.
While this information is in the "dictionary" it should be reflected in the class definitions.

See above.

In some cases it is and in others not. The inconsistency
makes review difficult. The lack of enumerated values (even in the "dictionary") means that the specification is incomplete.

Clarification is needed. Incomplete for what purpose?

8.9 DO_Child_Data
The class description does not describe a DO_Child_Data class. It should be more specific.

***1 Further discussion needed on the use of abstract classes for categorizing.

Same is true for 8.10 DO_Core_Data, 8.11 DO_Core_Other

All the DO_* classes are missing Cardinality values for attributes.

There are no attributes.

8.15 Document
The hierarchy of
Data_object_Description->DO_Taggable->DO_Core_Other->Document is overly abstract.
The DO_* classes have no attributes so contribute nothing to the definition of the Document class.

***1 This is one viewpoint. Another is that regardless of its complexity, the conceptual class hierarchy should represent the domain as well as possible. Granted that for PDS3 this has been more difficult than usual. This particular complexity more likely represents anomalies that should be addressed in PDS4.

** General Comment **
The DO_* classes make the model more complex than is necessary.

8.18 File_Explicit
The attribute "record_type" is an enumeration, but no values are indicated in the "Value" column.
Same is true for 8.19 File_Implicit.

>>ENUM

8.20 Gazetteer_Column
Where did Gazetteer_Column come from? Its not in the PSDD. Gazetter_Table uses Column.

This resulted from a decision to model required named columns distinct from user named columns.

8.22 Header
Attributes "header_type" and "interchange_format" are enumerations.

>>ENUM

8.23 Histogram

8.24 History
PSDD is the only attribute and its optional. Which means a valid History object can be empty. I don't think this is true in practice. This is an anomaly or can be fixed by looking at the catalog requirements for History.

History object previous captured as an anomaly in 011_080516_022_MG_1_History.

8.26 Index_Column
Other values for column "name" are possible. The list in "Value" indicate only those listed are allowed. The specification does not clearly describe this.

***1 Clarification needed. The has_Column association seems to address you concern.

8.27 Index_Table
Uses "not_applicable_constant" and "unknown_constant" while Table uses "missing_constant" and "invalid_constant". Consist names should be used. Should this be an anomaly.

***1 Good question. Should something like the union of the two sets should be proposed for both.

8.29 Palette
Palette is a sub-class of Table with some optional elements not used. Shouldn't this be modeled as such. It is what the PSDD states.

A decision was made to model Palette and several other Table subclasses as classes since PDS3 does not have a proper base Table class that could be used as the superclass. Hopefully this will be fixed for PDS4.

8.30 Parameters

PSDD is the only attribute and its optional. Which means a valid Parameters object can be empty.

Correct.

8.31 Qube
This is not a faithful capturing of the PDS3 spec. The PSDD does not show any sub-objects. Also the 8.32 Qube_Band_Suffix, 8.33 Qube_Line_Suffix, 8.34 Qube_Sample_Suffix and 8.35 Qube_Suffix are not PSDD objects.

The QUBE had been previously reviewed and updates made.

8.32 Qube_Band_Suffix
Description does not match class.

8.33 Qube_Line_Suffix
Description does not match class.

8.34 Qube_Sample_Suffix
Attribute "sample+suffix_name" has a typo "+" => "_"
Description does not match class.

8.35 Qube_Suffix
Description does not match class.

8.36 SPICE_Kernel
Description does not match PSDD. Type in last sentence "Also part of SPICE of the SPICE". Strike first "of SPICE"

>>WCDC

8.38 Series
Attribute "table_storage_type" is not part of Series object.

***1 Correct. This results from the decision to model Series as a type of Table and is evidence of the problem of PDS3 not having a good base Table class. The fix given PDS3 standards is to model Series as a class at the same level as Table.

8.38 Spectrum
Attribute "table_storage_type" is not part of Series object.

Ditto

8.43 Table
Other subclasses include Plalette, Index_Table, Gazetteer_Table.

See above.

9.3 IDE_Ancillary
IDE_Ancillary is identical to the class Identification_Data_Elements except for name. This is redundant and unnecessary.

***1 This is one viewpoint. The other is that IDE_Ancillary is simply a proper subclass of Identification_Data_Elements. This modeling approach demonstrates how the PDS model can address (to some small degree) the issue of optional attributes.

9.4 IDE_Earthbase
IDE_Earthbase is identical to the class Identification_Data_Elements except for name and restrictions on the use of attributes which are not explained and are not needed. This is redundant and unnecessary.

9.4 IDE_Spacecraft
IDE_Spacecraft is identical to the class Identification_Data_Elements except for name and restrictions on the use of attributes which are not explained and are not needed. This is redundant and unnecessary.

9.8 TDO_Core
This is all that is needed to describe a tagged data object. A Data_Object_Description is polymorphic, with Data_Object and Pointer generic.

***1 Please define your terminology [polymorphic, generic] for this context.

Describing every permutation is unnecessary. That is, all the "Tagged_*" are uncessary.

***1 A decision was made to model the various types of tagged_data_object. This is consistent with the data modeling formalisms chosen.

The Inherited Association "has_Data_Object_Description" has an "Ind" or "OR" is it optional or restricted. Can it be both?

***1 It is both. The association has_Data_Object_Description is first defined for tagged_data_object as optional with 1 to many values. In the subclass TDO_Supplemental the inherited association remains optional and has the allowed classes restricted.

There does needs to be some cleanup to make the superclasses consistent with their leaf subclasses.

9.13 Tagged_Data_Object
Unnecessary (See 9.8 TDO_CORE comment)
Cardinality for associations needs to be 1, 1, 1 and "Ind" is "R".

***1 Good call. This is some of the cleanup mentioned above.

9.14 Tagged_Document
Unnecessary (See 9.8 TDO_CORE comment)
Cardinality for associations needs to be 1, 1, 1 and "Ind" is "R".

Will be fixed via inheritance through 9.13 fix.

9.13 Tagged_File_Implicit
Unnecessary (See 9.8 TDO_CORE comment)
Cardinality for has_Pointer be 1.

9.13 Tagged_File_Implicit_Document
Unnecessary (See 9.8 TDO_CORE comment)
There is no has_Pointer in an implicit file. Perhaps that is what card of

"none" is suppose to indicate.

Yes

Its unclear.
9.20 Tagged Header
Unnecessary (See 9.8 TDO_CORE comment)
Cardinality for has_Pointer be 1.

Will be fixed via inheritance through 9.13 fix.

9.27 Tagged_SPICE_Kernel
Unnecessary (See 9.8 TDO_CORE comment)
Cardinality for has_Pointer be 1.

9.29 Tagged_Software
Unnecessary (See 9.8 TDO_CORE comment)
Cardinality for has_Pointer be 1.

9.34 Tagged_Text
Unnecessary (See 9.8 TDO_CORE comment)
Cardinality for has_Pointer be 1.

10.5 Product
The inclusion of "END" as an attribute is bizarre. This is a reflection of an ODL implementation. All classes have an extent which is represented differently in each implementation. What also makes "END" a bizarre attribute is that it is a sentinel in ODL and must occur at a specific location (at the end if the description). END should be removed and this change reflected throughout Chapter 10.

***1 I agree but needs WG discussion. Remaining product classes will be fixed by inheritance after fix of Product class.

11.2 Data_Set_Release
Data set release information is not part of a dataset, its external information about the data set. The Association of "part_of_data_set" should be removed.

The association "part_of_data_set" simply relates data sets and their releases.

11.4 Inventory_Data_set_Info
The use of "has_Data_Set_Inventory" is ambiguous. In nearly all other instances "has_" indicates composition. In this case it's a reference (?). Perhaps it should be "refers_to-"

***1 As its definition in the data dictionary states, it is a simple association.

***1 Needs discussion. - There is an issue related to types of associations and "refers_to" is a better name.

11.5 Inventory_Node_Media_Info
"medium_type" is an enumeration.

>>ENUM

Chapter 12
Remove entire chapter. The text doesn't relate to the PDS3 specification.

***1 The text does need to be updated. However this chapter provides the "formal" basis for the "tagged_data_object" concept.

Chapter 13
The introduction needs to be re-written. It misstates many things.

Done.

Throughout chapter 13 there are references to data types. These types are not defined. For example what does a type of "character" mean? One character or a string of characters? I know it's the formerly, but it should be clearly defined.

data types to be added to the dictionary.

There should some introductory text that explains the general format of a definition. For example, does "value" mean it always has this value, or one of the values in the comma separated list.

The latter. The cardinality is provided in the model where the attribute is used.

** General Comment **
The specification is incomplete without listing all the allowed (enumerated) values.

The specification is a published version of the Information Model database for a particular audience, in this case reviewers. The Information Model database has the entire content of the PSDD. If the Information Model is to be used for implementation, the complete lists of valid values are available as enumerated lists or any construct specified by the target language.

WG agreed to publish the document, after review is complete, with all valid value listed.

Type of "context_dependent" is very vague. There are two instances where "context_dependent" is used. The first is where a unitized value is expected (real + units). The second is where the value can be either an integer or non-decimal. This is really just one data type that can be represented in different ways.

CODE:>>DDICT - This code consolidates most of the remaining data dictionary issues. a) In general the PSDD information has been captured in the specification as is. This is consistent with the purpose of the specification model which is to capture the PDS3 model as it is currently documented. Issues associated with the current model, especially where the standards are clear, are not applicable for this review. b) Since the specification dictionary is a subset of the PSDD and its purpose is to support the information model, there is no need to present all data element definition information such as min/max values, standard value types, or formation rules. The PSDD can be referenced for such information. When and if it is used for driving an implementation this information, which in the info model database, can be presented. c) WG consensus is that context_dependent is well defined and used consistently in PDS3. d) The name of the data dictionary is

being changed to specification dictionary. It will contain attribute, association and some valid value definitions.

13. axis
Limit to range of values. Must be 1->n.

>>DDICT

13. axis_Interval
Type is always real + units.

>>>DDICT

13. axis_items
Limit to range of values. Must be 1->n.

>>>DDICT

13 band_sequence
Underscores precede value names. (Typo)

***1 Good call.

13 bit_data_type, core_item_type, data_format, data_object_type, data_type, header_type, index_type,
instrument_host_id, instrument_id, kernel_type, line_display_direction, medium_format, facility_name,
instrument_host_name, instrument_name, mission_aliase_name, mission_name, pds_version_id, platform,
positive_longitude_direction, product_type, sample_display_direction, sample_suffix_valid_minimum,
software_license_type, software_purpose, suffix_item_type, target_type, technical_support_type,
volume_format,
Data type is character. A type of "identifier" implies a specific formation

rule which doesn't apply
to the allowed values.

>>>DDICT

13 c_axis_radius, center_latitude, center_longitude, derived_maximum, derived_minimum,
  maximum, minimum, minimum_sampling_parameter, offset, scaling_factor, valid_minimum,
  valid_maximum
Data type is real + units. Just "real" may suffice as long as "real" is
defined to be a potentially unitized value.

>>>DDICT

13 collected_about, collect_by, collected_in, collected_on, distributed_by,
all has_* terms
Are not data dictionary terms. They are labels for associations used in the
model.

>>>DDICT

13 curated_by, description
Add Type: character

Curated_by is an association.

>>>DDICT

13 curates
Add Type: indentifier (? Or character)

>>>DDICT

13 data_set_collection_id, data_set_collection_name, data_set_id, data_set_name
  volume_series_name, volume_set_id, volume_set_name,

While this is an "enumeration" allowed values should not be listed since the list is out of date as soon as a new datasets or products are generated.

>>>DDICT

Change type to "character"

>>>DDICT

13 file_name
Change type to "file_name". It has a specific limited character set and formation rule so a new data type is warranted.

>>>DDICT

13 inode_nstitution_name
Typo. Should be node_institution_name

Previously fixed.

13 invalid_constant
Strange constraint in the text. How is this represented in the model?
Type is real+units

>>>DDICT

13 line_high_instr_sat, line_highe_repr_sat, line_low_instr_dat, line_low_repr_sat, line_suffice_*, sample_*
There are no definitions. Presumable because this do not exist in PDS3 and were a result
Of the new Qube classes which should be removed.

>>>DDICT - Definitions could be added if provided.

13 mission_start_date, mission_stop_date
Change type to "date_time" or something similar. "date" and "date+time" are different data types.

>>>DDICT

13 not_applicable_constant, unknown_constant
Can be either real or character. An anomaly. Perhaps "character" is sufficient.

>>>DDICT

13 pds_address_book_flag
Where is used.

Yes, 7.18 Personnel.

Is "null" really allowed?

This is what the PSDD indicates.

13 product_id, resource_id
Data type: identifier

>>>DDICT

13 required_storage_bytes
Data type: integer

>>>DDICT

13 resource
Remove, not used.

This is an association, see 11.1 Data_Set_HouseKeeping.

13 resource_link
Change type to "url". It has a specific limited character set and formation rule so a new data type is warranted.

>>>DDICT

13 suffix_high_instr_sat, suffice_high_repr_sat, suffix_low_instr_sat, suffix_low_repr_sat,
Suffic_null, suffix_valid_minimum,
Change data type to integer. Define integer to allow different expression, decimal, bit field, etc.

>>>DDICT

14 Cardinality
The definition is correct, usage of cardinality in class definitions do not adhere to this definition.

***1 Please clarify and give specific examples.

15 010_080204_001_DataSetProductMap
This is not an anomally. This is part of what we want.

***1 This is an anomaly since it is not clear that there is a node consensus on allowing a product to belong to more than one data set.

The second half the description isn't correct. A product can be associated with multiple datasets because a set of dataset_ids can be given. We do this at PPI.

Also SBN

-Todd-

16. PDS Specification - B. Semenov Review 1 - Response 2 - Final - SPICE

Hi all,

The following includes the final responses to B. Semenov's review of the specification document.

thanks,
steve

080807 - Review of responses - WG (AR, MG, SH, RJ, CI, EG, LH)
080723 - Response to review comments - Steve
080723 - Review comments - Boris

Review of responses - 080807
---

1) The WG consensus is that the File_Explicit class definition in the specification document is consistent with the definition in the PDS3 standards. In addition the inclusion of "Spice_Kernel" as an optional class is appropriate. "Spice_Kernel" is simply allowed, whether or not it is being used, and furthermore suggests that Spice_Kernel is "first_class" or equal to other objects in this regard. The same opinion holds for "SPICE" as a possible value for data format.

2) The WG suggests omitting the word "only" in the proposed definition. -- "SPICE Kernel is a data file containing navigation and other ancillary data. Different types of SPICE kernels store different kinds of navigation and ancillary information. Some kernels are binary files while the others are text files. SPICE kernels are used only in conjunction with SPICE Toolkit software to compute observation geometry."

3) ***2 An anomaly will be added (stated as an issue) that NAIF requests PSDD be removed from the SPICE_Kernel class definition.

4) The WG consensus is that the current model for the "Data Product" class, where Tagged_Data_Objects (TDOs) are subordinate to the Tagged_File_Implicit (TFI) class is correct. Furthermore, the name of the association between Data Product and Tagged_File_Implicit, "has_TFI_TDO" is correct. The definition of the association has_TFI_TDO will be expanded to describe the subordinate relationship. The introduction to Section 10 - Product will be expanded to describe how one navigates a path from the product class to specific tagged_data_objects. (e.g. IMAGE, TABLE)

5) ***2 An anomaly will be added (stated as an issue) to address multiple instances of "SPICE_Kernel" in the PSDD valid values list.

6) The question as to what data sets use various values can not be answered systematically at this time.

NAIF Notes on PDS Information Model Specification
(DRAFT Version 070916q, HTLM format).

-- (Fig 1) this figure is the only place in the document that
   mentions SPICE_Product (as well as Ancillary_Product,
   Document_Product, and Software_Product), probably because
   it is out of sync with the rest of the document.

The UML diagrams are lagging the specification and will be updated.

-- (8.18) why can File_Explicit have Spice_Kernel?

***2 Discussion needed but NAIF feels that File_Explicit should not include Spice_Kernel as an optional PDS object.

-- (8.36) replace description of SPICE_Kernel with the following text:

"SPICE Kernel is a data file containing navigation and other ancillary data. Different types of SPICE kernels store different kinds of navigation and ancillary information. Some kernels are binary files while the others are text files. SPICE kernels are used only in conjunction with SPICE Toolkit software to compute observation geometry."

Done.

-- (8.36) remove PSDD from SPICE_Kernel; no keywords other
   than the three required ones should ever appear in this object

***3 Remove PSDD from SPICE_KERNEL.

-- (9.32) change Pointer to Data_Object_Pointer

Done.

-- (9.4)(8.19) why should both Descriptive_Data_Elements
and Implicit_File allow PSDD?

***1 Needs discussion. However, any PSDD elements in Implicit_File would describe the FILE. Any PSDD elements in Descriptive_Data_Elements would describe the product.

-- (10.1) why Data_Product has_TFI_TDO instead of has_TFI and
has_TDO? Having them both (as two separate lines) will help
tracing down to SPICE_Kernel easier.

***1 Needs discussion. The current model has TDOs as children of the Implicit File class. The suggestion would make them siblings. It would simplify the model by not requiring specialization of the Implicit File class if data product is specialized.

-- (10.1) hard to follow path from Data_Product to
SPICE_Kernel because TDO does not appear in Data_Product
associations

***1 Do we need to specialize data product to include core types, e.g. IMAGE, SPICE_KERNAL, TABLE. Also see has_TFI_TDO item above.

-- (13) SPICE_Kernel is listed twice in data_object_type value
list (from PSDD)

***3 - The PSDD includes the term SPICE_Kernel twice, once with underscore and once with blank. For the specification, the blank is replaced with two underscores. (Check)

-- (13) data_format has "spice" as one of possible value (from PSDD).
I was not aware that data_format was a way to "identify" SPICE
data. Is it possible to find out which data sets that used
this keyword for this purpose?

***3 - The PSDD includes "spice" as a possible value for data format.

-- (13) product_type value list (from PSDD) includes a strange
mix of SPICE related values: SPICE_SP_KERNEL, BCK, BSP , ENB,
SAK, SCK, SPC, SPICE_KERNEL, SPICE_SP_KERNEL, SPK. Is it
possible trace them to the source data sets?

***3 - The PSDD includes a strange mix of SPICE related values.

-- (general) it is very difficult (for me) to review a document
with so many TBDs

***1 General cleanup of descriptions is needed.

-- (general) having a better idea on how document is
organized and what various things in it mean (thanks to Steve
for a tutorial and Q&A session), I feel that it seems to do
a much better job at describing relationships and uncovering
anomalies in PDS3 than STDREF and PSDD.

---

17. PDS3 Specification - T. King Review 1 - Follow Up Question - Response 1.a - Final

Hi all,

The following includes the final response to T. King's answer to the cardinality question.

thanks,
steve

080807 - Review of answer - WG (AR, MG, SH, RJ, CI, EG, LH)

Review of responses - 080807
---
1) The WG consensus is that the use of the "IND" (Indicator) column and the value "O" (Optional) to indicate "optional" attributes (i.e. the possibility of "no attribute") is effective and does not need to be changed.

2) The definition of cardinality will be changed to be PDS3 specific.

"Cardinality" is the number of values allowed to an attribute or association in a single class. Cardinality in general is stated as a range with a minimum and maximum. For example, an attribute that may be multi-valued will have a cardinality of "1..*". A cardinality where the minimum and maximum are the same is often shown as the single value. For example, an attribute required to have exactly one value will have a cardinality of "1". When a value is required the minimum cardinality is at least 1. At least one value is always required in PDS-3."

3) The WG recognizes that during implementation the strict modeling formalisms may be relaxed. For example, in PDS3 the case of "no value" is not allowed in a syntactic sense, since if an optional attribute is included in a label, it must be assigned a <token>. So for example, it is reasonable that in the generation of an XML/Schema where the construct minOccurs="0" means "no attribute" (and an implied "no value") the construct "minimum cardinality = 0" could be used in the UML modeling tool.

4) In summary, the PDS3 Information Model attempts to capture the PDS3 model in an unambiguous and implementation independent manner.

>
> 14) Request Todd to clarify and provide examples for "14 Cardinality
> - The definition is correct, usage of cardinality in class
> definitions do not adhere to this definition."
>
>

Search Header:king
Anywhere:PDS3 Specification - T. King Review 1

Hi Steve -

Here's the example to illustrate my point for this comment. I'm sending it as HTML so I can embedded figures to help the discussion:

> >Another point about allowing "0" as a minimum cardinality is
> that with
> >current modeling environments (like Visual Paradigm) you can
> generate
> >an XML Schema based on the model, the XML Schema can then be
> processed
> >by JAXB to generate Java classes for a validating
> parser/generator. The
> >parser/generator can then be used in tools. This
> three-clicks to code
> >is a real aid to development. Maintaining the "optional" designation
> >outside the cardinality makes the three-clicks to code
> impossible since
> >more complicated processing must be done.
>
> This is an implementation issue. But please explain how your UML tool
> omits an attribute based on any combination of an attribute's min or
> max cardinality.

Allowing cardinality of "0" is instrumental in validating whether a data stream is well formed. The mere possibility that an attribute may exist means that a data structure must have a place to store it. I think this is where the "0" vs. "1" debate originates. One viewpoint is for validation of streams the other is for data structures (or database schema). If at the modeling phase you disallow "0" then you disallow validation unless you re-introduce the concept of optional in some other way. Since "0" is well defined and handled correctly in modeling tools why disallow it?

Here's an example to illustrate this point. Assume we have a simple class called "test' which is defined as:

If I click to generate Java I get:

```
public class Test {
 private Object _name;
 private Object _type;
 private Object[] _attribute;
}
```

If I click to generate C++ I get:

```
#include <string>
```

```
#include <vector>
#include <exception>
using namespace std;

#ifndef __Test_h__
#define __Test_h__

class Test;

class Test
{
 private: string _name;
 private: string _type;
 private: string* _attribute;
};

#endif
```

and the XML Schema looks like:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema
 targetNamespace=" http://Cardinality"
 xmlns="http://Cardinality "
 xmlns:xs=" http://www.w3.org/2001/XMLSchema">
 <xs:complexType name="Test">
  <xs:all>
   <xs:element name="name" type="xs:string" minOccurs="0" maxOccurs="1">
   </xs:element>
   <xs:element name="type" type="xs:string" minOccurs="1" maxOccurs="1">
   </xs:element>
   <xs:element name="attribute" type="xs:string" minOccurs="1"
maxOccurs="unbounded">
   </xs:element>
  </xs:all>
 </xs:complexType>
</xs:schema>
```

The XML Schema can be used for validation so if the class is stored in XML the following would validate:

```
<Test>
  <name>something</name>
  <type>something</type>
  <attribute>something</attribute>
</Test>
```

as would the following since name is optional (minOccurs="0")

```
<Test>
  <type>something</type>
  <attribute>something</attribute>
</Test>
```

or the following since "attribute" can have multiple occurrences:

```
<Test>
  <type>something</type>
```

```
  <attribute>some</attribute>
  <attribute>thing</attribute>
</Test>
```

Since ODL can be translated into XML it is possible to use the XML Schema to validate an ODL version. The translation could be a prefilter to the ODL stream and transparent to user since the XML would be an "internal" representation.

So, the following ODL would also be valid:

```
OBJECT=Test
  name=something
  type=something
  attribute=something
END_OBJECT=Test
```

as would:

```
OBJECT=Test
  type=something
  attribute=something
END_OBJECT=Test
```

and

```
OBJECT=Test
  type=something
  attribute={some, thing}
END_OBJECT=Test
```

Hopefully this illustrates the power of allowing a cardinality of "0".

-Todd-

18. PDS3 Specification - T. King Review 2 - Response 2 – Final - Polymorphic Question

Hi all,

The following includes the final response to T. King's answer to the polymorphic question.

thanks,
steve

080807 - Review of answer - WG (AR, MG, SH, RJ, CI, EG, LH)

Review of responses - 080807
---
1) The WG consensus is that the modeling of subclasses of Tagged_Data_Objects is effective and enlightening and does not need to be changed.

2) The WG recognizes that the need to explicitly model all subclasses may be relaxed during implementation especially in the case where no additional information is provided by the subclass definitions. However the concept of polymorphism can not be applied during implementation if a class hierarchy does not exist. The definition of a class hierarchy in a data model is much preferred to the encoding of a class hierarchy in the software.

3) The attribute "name" will be added to the class definition of data_object_pointer to address the <token> on the left side of the assignment statement.

At 08:21 AM 7/29/2008, Todd King wrote:
Hi Steve -

The short answer is that the model for TDO_CORE can be a list of allowed values. That is,

```
has_Data_Object_Description    1        Array
                               Collection
                               Document
                               File
                               (and so on)
has_Data_Object                 1
has_Pointer                1
```

Hence the term "polymorphic" since it can represent may different objects with a single pattern.

But the modeling of the "pointer" is still very awkward.

In PDS3, a pointer simply assigns a name to a bit string. This is accomplished by allowing both the left and right side of an assignment to be variable. On the left side of the assignment the characters that follow the "^" form the name and the right side is the specification of where the bit string is stored.

Also in PDS3 a name is assigned to an instance (OBJECT). It is through name

matching of pointer and instance that descriptions and bit strings as linked. We've often thought of the pointer as the nexus of the association, but a different perspective makes the conceptual model simpler. If we take an instance (OBJECT) perspective then we start with the description, then ask where are the bits that are described. If a search for a pointer with the same name as the instance exists, then it describes the bit string. If none exists then the bits are assumed to be in the same file as the description (attached label).

But this is all implementation. Conceptually its very simple. Each class describes the contents of a bit string. So each class should have a "has_Bit_String" association which would then eliminate all of Chapter 9.

-Todd-

> -----Original Message-----
> From: J. Steven Hughes [mailto:J.Steven.Hughes@jpl.nasa.gov]
> Sent: Friday, July 25, 2008 1:38 PM
> To: tking@igpp.ucla.edu
> Subject: Fwd: PDS3 Specification - T. King Review 2 - Response 2
>
> Hi Todd,
>
> In responding to your second review, the WG requests further
> explanation on the following item.
>
> thanks,
> steve
>
>
>
> >6) Ask Todd to expand comments on 9.8 TDO_CORE [polymorphic, generic].
> >
> >
> >>9.8 TDO_Core
> >>This is all that is needed to describe a tagged data object. A
> >>Data_Object_Description is polymorphic, with Data_Object and Pointer
> >>generic.
> >
> >***1 Please define your terminology [polymorphic, generic] for this
> context.
> >
> >
> >> Describing every permutation is unnecessary. That is, all the
> >>"Tagged_*" are uncessary.
>

19. PDS Specification - E. Guinness Review 1 - Response 2 - Final

Hi all,

The following includes the final responses to E.Guinness's review of the specification document.

thanks,
steve

080814 - Review of responses - WG (AR, MG, DS, SH, RJ, CI, EG, LH)
080813 - Response to review comments - Steve
080813 - Review comments - Ed

Review of responses - 080814
---
1) Update to the "best practice" for modeling cardinality of the relationship between an attribute and its values. I.e. The number of values allowed.

a) If an upper bound other than "no upper bound" is explicitly specified, then it will be used. (e.g. axis_name 1..6 BAND)

b) If the standards explicitly state that the cardinality has "no upper bound" then the cardinality will be presented as "1..*".

c) If the PDS3 standards do not specify the cardinality's upper bound then the cardinality will be presented as "1".

2) Fix the optional and required attributes error for Band_Bin. Check for other similar errors.

3) Fix both the file name and the offset attributes for Data_Object_Pointer to be required and not optional. The class description is to reference an anomaly. See below.

4) ***2 Data_Object_Pointer - In a PDS3 attached label, both file name and offset are not required. Address this issue together with the issue of attached labels.

5) ***2 Attribute Ordering - The implementation requirements on SFDU and PDS_VERSION_ID suggest that the model needs to specify attribute ordering, at least for these two attributes. For the PDS3 model specification this will not be addressed other than to state that the PDS3 standards need to be consulted for the particulars.

6) ***2 Units - <confirm anomaly exists> The PDS3 model specification does not provide "units". The modeling database used to generate the PDS3 model specification includes a copy of the PSDD. Units, min/max values and all other data element definitional information is available. The presentation of units will be taken up as part of PDS4.

7) ***2 Implicit Assumptions - In general, during the development of the PDS4 data architecture, implicit assumptions are to be made explicit where ever possible. (e.g. Histogram) - "How can the bits be interpreted without knowing if the values are ascii (unlikely case) or binary? Or how does the model capture assumptions or defaults like this is it assumed that the histogram is binary if interchange format is not listed? The description of histogram in the standards has other assumptions that the model does not capture. For example, if scale and offset are not given, then they are assumed to be 1 and 0.)"

8) ***3 Bytes is not listed as an element of Field in the on-line data dictionary. It is listed as such in the PSDD and Stds Ref.

9) ***2 The PSDD does not capture relationships between data elements such as "has similar meaning" and "has similar valid values".

10) ***2 Text - "Why do we have a text object? Shouldn't it be related to a document? Why does text use note (required) and document use description (optional) as attributes? Why is interchange format optional? Shouldn't text objects always be ascii? Even though EDCDIC is a standard value for interchange_format, would we allow a text object in EDCDIC or a binary text object?"

11) ***3 Inconsistency between PSDD and Std Reference to be sent to E. Rye. - "Image_map_projection: The spec lists several attributes as optional when the standards ref. says that are required first_standard_parallel, second_standard_parallel, reference_latitude, reference_longitude."

12) ***2 Compression - Compressed file objects are not handled consistently, especially in regard to legacy data. (e.g. Huffman First Difference) - Consider the subclassing of explicit file for a compressed file class in PDS4. -- "There is nothing in the spec about modeling compressed data, whereas the standards ref. has a separate appendix that lists requirements for describing some types of compressed data. The requirements for JPEG2000 and ZIP list two required objects that I don't see in the spec compressed_file and uncompressed_file (and uncompressed_file seems to require the image object). Requirements for other compression methods are not described in the appendix (listed as tbd), but existing usage in PDS does not conform to the way JPEG2000 and ZIP are described."

13) ***2 Browse_Images - "There seems to be an implicit rule in PDS that browse products cannot be described as images because they are compressed and not considered

by some to be data. Yet encoding_type is a valid optional attribute for the image object and its standard values includes several compression methods."

At 09:31 AM 8/13/2008, Edward A. Guinness wrote:
Steve,

Here are my comments on PDS3 spec. Sometimes I ask a question because the question seems to be an anomaly to me. Also, as one of the last reviewers, I am sorry if a raise an issue that has already be discussed and decided upon.

7.2 Data_object_pointer: Both attributes are listed as optional. Does that mean you can have a valid pointer that contains neither?

Yes, according to the model. If we want to preclude combinations that are not used, we should extend the model to include all POSSIBLE data_object_pointer subclasses and then allow the subclasses where needed.

7.10 Label_standards_identifiers: The description says that each PDS label must begin with PDS_VERSION_ID, but if an SFDU is present, the standards say that it must come first (std ref 5.3.1).

This is implied by defining the SFDU at the Product level but the issue seems more of an implementation detail. For example, if we ultimately define an XML version of a label, is the SFDU required at all. If allowed it could not be first since the XML headers are required to be first.

8.2 Band_bin: All attributes are listed as required, but the standard reference only lists 4 required attributes, the rest are optional.

Good call. Fix - Cardinality = "1" for assignment of a sequence and and have Optional indicator.

8.16 Field: Not a problem with the spec. But I noticed that bytes is not listed in the data dictionary as being required for field.

***1 I might be missing something but bytes seems to be required in the stds ref, the psdd, and the spec.

8.22 Histogram: Why is interchange format optional? How can the bits be interpreted without knowing if the values are ascii (unlikely case) or binary? Or how does the model capture assumptions or defaults like this is it assumed that the histogram is binary if interchange format is not listed? The description of histogram in the standards has other assumptions that the model does not capture. For example, if scale and offset are not given, then they are assumed to be 1 and 0.

***1 The model uses the PSDD as the gold standard so interchange_format is considered optional. This could be captured as several anomalies. 1) Interchange format should be required. 2) PDS4 model should allow default specifications. 3) Model needs to capture the additional assumptions.

8.24 Image: This object uses several attributes that are specially named for the image object, but where there are other keywords that have similar meaning and that used in most other objects: sample_type vs data_type, sample_bits vs bytes, and for the description attributes image_id vs product_id.

***2 Anomaly - The PSDD does not capture relationships between data elements such as "has similar meaning" and "has similar valid values".

8.39 Text: Why do we have a text object? Shouldn't it be related to a document? Why does text use note (required) and document use description (optional) as attributes? Why is interchange format optional? Shouldn't text objects always be ascii? Even though EDCDIC is a standard value for interchange_format, would we allow a text object in EDCDIC or a binary text object?

***2 These all seem to be good candidates for anomalies.

11.9 Data_set_map_projection: The standard reference says that this object has one required attribute and no optional ones and requires the data_set_map_projection_info object that contains the attributes listedin the spec as optional.

When ODL uses an Object to simply group keywords together and when the Object can be collapsed without a loss of modeling information, it is collapsed and the cardinalities of the contained data elements are inherited from the Object.

11.13 Image_map_projection: The spec lists several attributes as optional when the standards ref. says that are required first_standard_parallel, second_standard_parallel, reference_latitude, reference_longitude.

***3 Inconsistency between PSDD and Std Reference to be sent to E. Rye.

Compression: There is nothing in the spec about modeling compressed data, whereas the standards ref. has a separate appendix that lists requirements for describing some types of compressed data. The requirements for JPEG2000 and ZIP list two required objects that I don't see in the spec compressed_file and uncompressed_file (and uncompressed_file seems to require the image object). Requirements for other compression methods are not described in the appendix (listed as tbd), but existing usage in PDS does not conform to the way JPEG2000 and ZIP are described.

***1 Good point for discussion.

Browse: There seems to be an implicit rule in PDS that browse products cannot be described as images because they are compressed and not considered by some to be data. Yet encoding_type is a valid optional attribute for the image object and its standard values includes several compression methods.

***2 Anomaly ?